

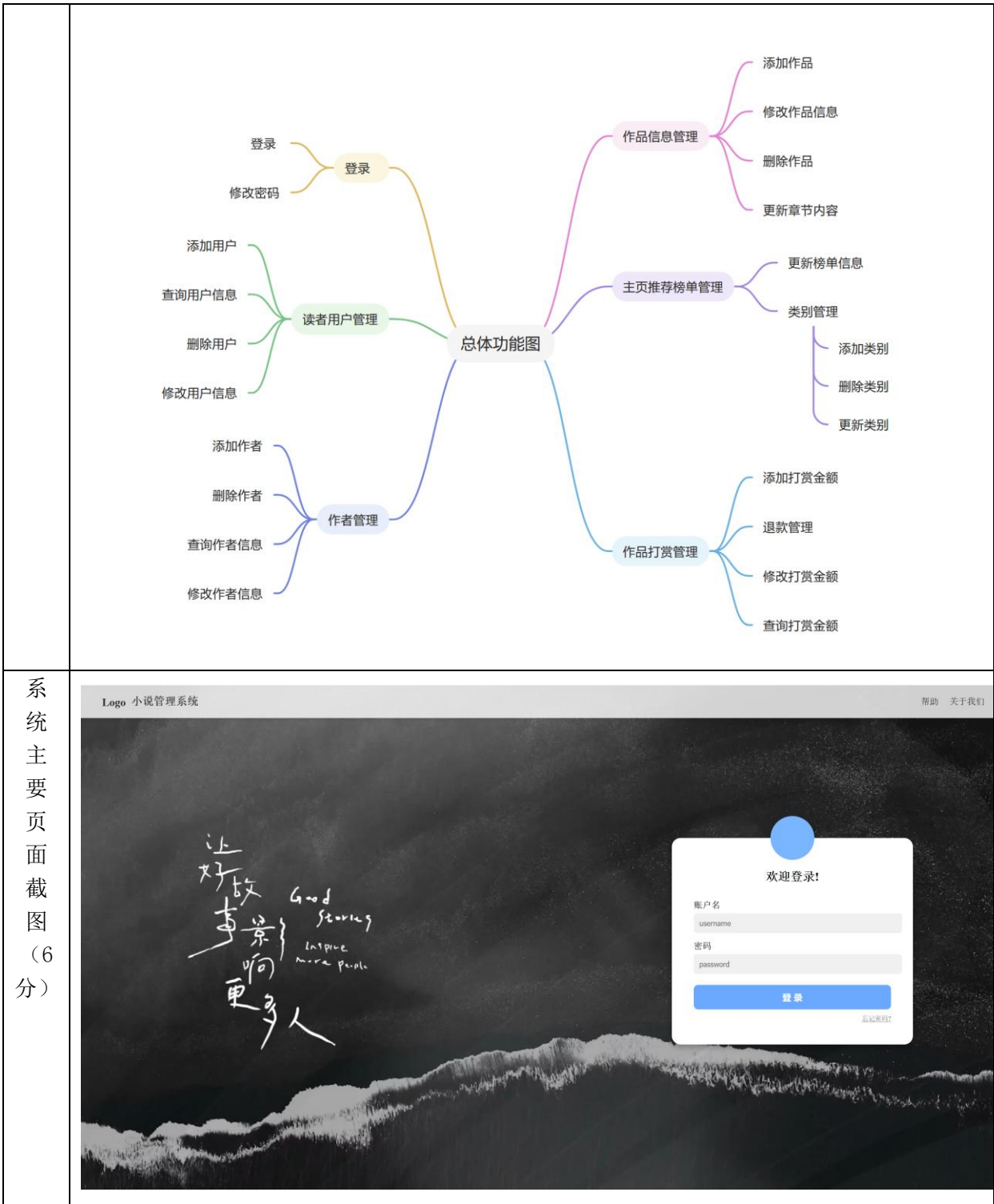
数据库工程作业

- 要求:
- 1. 完成一个小型的数据库信息管理系统（或部分功能），并填写工程作业报告；程序和报告请在规定时间之内上传。
 - 2. 开发模式（B/S 或 C/S）、开发高级语言任选，后台数据库使用大型数据库管理系统（SQL Server、Oracle、MySQL 等），**不要使用桌面数据库。**
 - 3. 报告中所列举的四种操作，**每种操作举一个例子即可。**
 - 4. 作业成绩按照报告中的标准评分，**程序只实现报告中涉及的部分即可。**
 - 5. 作业完成后，**请将工程作业报告和程序打包提交给助教老师，并联系助教老师进行系统说明和演示，回答相关问题。**

工程作业报告

1. 项目信息（10 分）

学号	2213230	姓名	向宇涵	专业	计算机科学与技术
项目名称	小说阅读网页管理系统				
必备环境	1. 前端：React（Javascript） 2. 连接：Python（Trae） 2. 后端数据库：mysql (navicat)				
系统主要功能简介（4分）	小说阅读网页管理系统旨在提升小说网页的管理效率和现代化水平，通过该系统，小说网页可以实现对阅读用户、作者信息、作品推荐的管理，提高管理效率，为读者提供更好的小说推荐服务。实现了登录、读者用户管理、作者管理、作品信息管理、主页推荐榜单管理和作品打赏管理六大功能。具体的功能如下图所示：				



Logo 小说管理系统

帮助 关于我们

让好故事影响更多人

Good Storying
Inspire more people

欢迎登录!

用户名

密码

登录

忘记密码?

系统主要页面截图（6分）



盛世美颜巅峰榜

根据热度评分、人气、互动综合分排行

01



十日终焉

杀虫队队员

悬疑惊悚

02



我不是戏神

三九音域

都市异能

03



我在精神病院学斩神

三九音域

悬疑惊悚

04



无限末世：开局一座地窖安全屋

一颗萝卜

科幻末世

05



癫，都癫，癫点好啊

小盐子

娱乐星光

06



攀高枝

白鸢成双

古代言情

管理功能



读者管理

管理读者信息



作者管理

管理作者账户



推荐榜管理

榜单排名管理



作品管理

作品信息维护



打赏模块

打赏金额管理

最新资讯

- 网络作家 | 盛世美颜小涵的原创
- 2025年5月盛世美颜小说全站与征稿活动公告
- 创作'富'攻略 | 掌握8个创作秘籍，轻松...
- 【课程回放攻略文章】让新手变高手...
- 「走进古风」看古风！主角配文...
- 这些院校，想做科幻学子的全金！
- 「青春校园」活动获奖公示



管理员

读者日志

退出

读者管理

请输入读者ID



搜索读者



添加读者

读者ID

用户名

密码

个性签名

0

等级

邮箱



管理员

作品日志

退出

作品管理

请输入作品ID

搜索作品

+ 添加作品

作品ID

作品名称

排名

分类

等级

作者ID

作者名称



管理员

推荐日志

退出

推荐管理

请输入作品ID

搜索推荐

+ 添加推荐

作品ID

作品名称

排名

分类

等级

作者ID

作者名称

管理员

作者日志退出

作者管理

2311366

搜索作者添加作者

作者ID

笔名

密码

作者ID: 2311366

笔名: 莫涵

账号密码: 123456

作品数量: 1

删除作者

管理员

打赏日志退出

打赏管理

请输入作品ID

搜索打赏添加打赏

作者ID

作品ID

作品名称

打赏类型

0

2. 系统配置（10 分）

说明		(2 分) 请说明系统配置情况 (后台数据库, 高级语言); (8 分) 请使用连接串连接高级语言和数据库, 并分析字符串的各个部分。			
配置 步骤 2 分	DBMS	1. Mysql 8.0.41			
		2. Navicat 16.3.7			
	高级 语言	1. Python 3.12.8			
		2. Trae 1.2.7			
		3. Node js 20.18.2			
连接串 分析 (6 分)		序 号	名称	功能说明	取值
		1	host	数据库所在的服务器地址,这里设置为 localhost,表示本地计算机的服务器	localhost
		2	port	数据库的端口号, Mysql 的端口号默认 3306	3306
		3	user	用于登录数据库的用户名	root
		4	password	用户的登录密码	Mysql0406!@
		5	database	需要连接的数据库的名称	nobelldb
		6	charset	数据库使用的字符集	utf8mb4
连接串代码 (截屏) (2 分)		<pre># 连接数据库 db = pymysql.connect(host="localhost", port=3306, user="root", password="Mysql0406!@", database="nobelldb", charset="utf8mb4")</pre>			
备注					

3. 数据库设计（14 分）

说明		（10 分）按照数据表的创建顺序，依次给出所涉及数据表的信息，其中参照字段以“（字段 1，字段 2，……，字段 n）”的形式给出，被参照字段以“表名（字段 1，字段 2，……，字段 n）”的形式给出； （4 分）一般 DBMS 都可以为数据库生成关系图，请将该图片截屏并粘贴到表格中。			
数据表 (10)	创建 顺序	数据表名称	主键	参照属性	被参照表及属性
	1	staff(管理员表)	staff_id(编号)	无	无
	2	reader(读者表)	user_id(账号)	无	无

	3	author（作者表）	author_id(作者 ID)	无	无
	4	works（作品表）	work_id(作品 ID)	author_id、 authorname	author(author_id)、 author(authorname)
	5	reward（打赏表）	author_id、work_id	author_id、 work_id、 authorname	author(author_id)、 works(work_id) 、 author(authorname)
	6	recommend(推荐榜)	work_id(作品 ID)	work_id 、 author_id、 authorname	works(work_id) 、 author(author_id)、 author(authorname)
	7	author_log(作者更改 日志)	log_id(修改 ID)	author_id	author(author_id)
关系图 (4)	<pre> erDiagram reward --o{ works : "author_id" reward --o{ works : "work_id" works --o{ author : "author_id" works --o{ author : "authorname" works --o{ recommend : "work_id" works --o{ recommend : "workname" recommend --o{ author : "author_id" recommend --o{ author : "authorname" reader --o{ author : "author_id" reader --o{ author : "authorname" author_log --o{ author : "author_id" staff --o{ author : "author_id" staff --o{ author : "authorname" staff --o{ author : "password" staff --o{ author : "writewords" staff --o{ author : "worknum" staff --o{ author : "level" work_detail_view --o{ works : "work_id" work_detail_view --o{ works : "workname" work_detail_view --o{ works : "category" work_detail_view --o{ works : "score" work_detail_view --o{ works : "pubtime" work_detail_view --o{ works : "introduction" work_detail_view --o{ works : "author_id" work_detail_view --o{ works : "authorname" </pre>				
备注					

4. 含有事务应用的删除操作（13 分）

说明	<p>(1 分) 简要说明该操作所要完成的功能;</p> <p>(2 分) 该操作会涉及的表 (必须含有两张或两张以上的关系表, 同时以“表名”的形式给出)</p> <p>(1 分) 表连接涉及字段描述 (描述方式为“表 1. 属性=表 2. 属性”)</p> <p>(1 分) 删除条件涉及的字段描述 (以“表名. 属性=? ”形式给出)</p> <p>(4 分) 实现该操作的关键代码 (高级语言、SQL), 截图即可; (其中如果删除语句中不包含任何形式的事务应用将扣除 3 分)</p> <p>(4 分) 如何执行该操作, 按所述方法能够正常演示程序则给分。</p>	
功能描述 (1 分)	<p>注销: 该操作用于注销自己的账号, 包括读者用户注销和作者解约注销账号。操作会在 reader 或 author 表中删除相应的信息, 如果是注销作者信息, 就需要在 works、reward 和 recommend 表中删除相关的作品信息。</p>	
涉及的表 (2 分)	<ul style="list-style-type: none"> • reader (读者用户信息表) • author (作者信息表) • works (作品表) • reward (打赏表) • recommend (推荐榜单表) 	
表连接涉及字段 (1 分)	<p>author.author_id=works.author_id</p> <p>author.author_id=reward.author_id</p> <p>author.author_id=recommend.author_id</p>	
删除条件字段描述 (1 分)	字段	规则
	reader.user_id=?	读者信息删除: 输入已有的读者用户的信息, 删除相应的用户所有信息
	author.author_id=? AND works.author_id=? AND recommend.author_id=? AND reward.author_id=?	作者信息删除: 输入已有的作者账号信息, 再在作品表和推荐榜单表以及打赏表单中查看是否存在此作者信息, 接着需要连着一一起删除

代码
(4分)

```
# 删除读者
@app.route("/api/reader/<user_id>", methods=["DELETE"])
def delete_reader(user_id):
    cursor = db.cursor()
    cursor.execute("DELETE FROM reader WHERE user_id = %s", (user_id,))
    db.commit()
    return jsonify({ "success": True, "message": "已删除" })
```

```
@app.route("/api/author/<author_id>", methods=["DELETE"])
def delete_author(author_id):
    cursor = db.cursor()
    cursor.execute("DELETE FROM author WHERE author_id = %s", (author_id,))
    db.commit()
    return jsonify({ "success": True, "message": "作者已删除" })
```

程序演示
(4分)

首先在读者用户信息注销时，在输入框中输入需要删除的用户账号，下面会显示出是否需要删除相关信息。

管理员

读者日志退出

读者管理

搜索读者

添加读者

读者ID

用户名

密码

个性签名

0

等级

邮箱

读者ID: 2311366

用户名: 莫涵

个性签名: 想天想地想宇宙

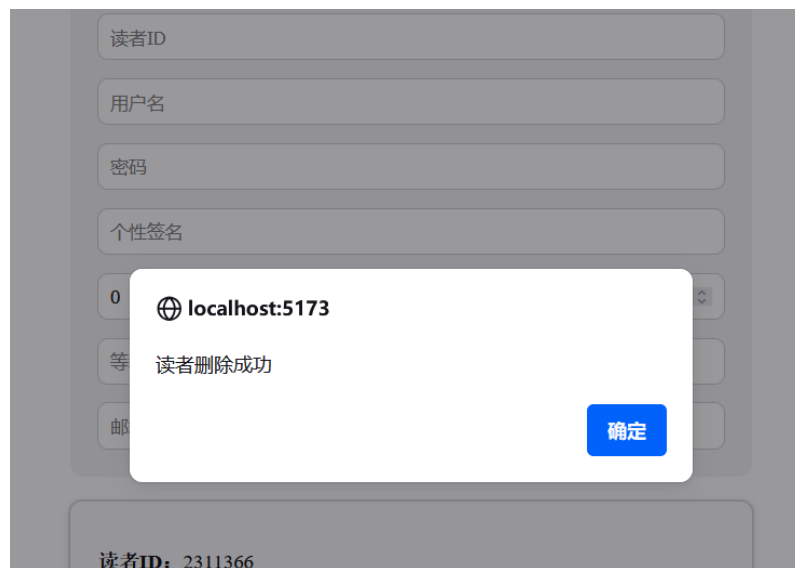
阅读时长: 121小时

等级: 小白

邮箱: 2311366@nankai.com

删除读者

点击删除，此时读者信息已被删除。



在作者管理页面，首先在输入框中输入作者的账号，查询作者账户的相关信息，点击删除作者，就可以把相关作者信息一并删除。



在其他模块查看此作者写的作品是否还存在，发现都被删除了。

	<div><div>1</div><div><div>搜索作品</div><div>+ 添加作品</div></div><div><div><div>作品ID</div><div>作品名称</div><div>类型</div><div>评分</div><div>作者ID</div></div><div><div>localhost:5173</div><div>没有此作品</div><div>确定</div></div></div></div>
备注	

5. 触发器控制下的添加操作（20 分）

说明	（1 分）简要说明该操作所要完成的功能； （2 分）简要说明该触发器所要完成的功能 （1 分）该操作会涉及的表（以“表名”的形式给出）。 （2 分）该操作输入数据以及输入数据应该满足的条件，如：数值范围、是否为空； （6 分）实现该操作的关键代码（高级语言、SQL），截图即可； （8 分）如何执行该操作，按所述方法能够正常演示程序则给分。	
功能描述 (1 分)	该操作用于在数据库中添加一个作者信息，该操作会将新的作者信息插入到 author 表中。	
触发器描述 (2 分)	该触发器用于在插入新的作者信息之后，对相关的表的数据进行更新和验证，确保数据符合一致性和完整性。这里可以创建一个触发器来记录每次添加的作者的操作日志。	
涉及的表 (1 分)	author	
输入数据 (2 分)	字段	规则
	author_id	作者 id 不可以为空且不能重复，且 id 长度不能超过 50 个字符
	authorname	作者姓名不能为空
	password	密码不可以为空，且密码的长度不可以少于 6 个字
	writewords	默认为 0
	worknum	默认为 0
插入操作源码 (3 分)	<pre> # 添加作者（带触发器支持） @app.route("/api/author", methods=["POST"]) def add_author(): data = request.get_json() author_id = data.get("author_id") pen_name = data.get("pen_name") password = data.get("password") writewords = data.get("writewords", 0) # 默认为0 worknum = data.get("worknum", 0) # 默认为0 # 确保数值字段为整数 try: writewords = int(writewords) if writewords else 0 worknum = int(worknum) if worknum else 0 except ValueError: return jsonify({"success": False, "message": "写字数和作品数必须为数字"}), 400 cursor = db.cursor() try: # 插入作者数据（触发器会自动执行验证和日志记录） cursor.execute("INSERT INTO author (author_id, authorname, password, writewords, worknum, level) VALUES (%s, %s, %s, %s, %s, '普通')", (author_id, pen_name, password, writewords, worknum)) db.commit() # 获取触发器执行后的日志信息 cursor.execute("SELECT details FROM author_log WHERE author_id = %s ORDER BY action_time DESC LIMIT 1", (author_id,)) log_result = cursor.fetchone() log_message = log_result[0] if log_result else "触发器执行完成" return jsonify({ "success": True, "message": "添加成功", "trigger_info": log_message }) </pre>	

	<pre> except mysql.connector.IntegrityError as e: db.rollback() if "Duplicate entry" in str(e): return jsonify({"success": False, "message": "作者ID已存在"}), 400 else: return jsonify({"success": False, "message": f"数据完整性错误: {str(e)}"}), 400 except mysql.connector.Error as e: db.rollback() # 检查是否是触发器抛出的自定义错误 if "SQLSTATE 45000" in str(e) or e.errno == 1644: return jsonify({"success": False, "message": f"数据验证失败: {e.msg}"}) , 400 else: return jsonify({"success": False, "message": f"数据库错误: {str(e)}"}), 500 except Exception as e: db.rollback() return jsonify({"success": False, "message": f"服务器错误: {str(e)}"}), 500 finally: cursor.close() </pre>
<p>触发器 源码 (3分)</p>	<pre> 1 -- 创建日志表（如果不存在） 2 CREATE TABLE IF NOT EXISTS author_log (3 log_id INT AUTO_INCREMENT PRIMARY KEY, 4 author_id VARCHAR(50), 5 action_type VARCHAR(20), 6 details TEXT, 7 action_time TIMESTAMP DEFAULT CURRENT_TIMESTAMP 8); 9 10 -- 设置分隔符 11 DELIMITER // 12 13 -- 创建作者添加前的触发器（数据验证） 14 CREATE TRIGGER before_author_insert 15 BEFORE INSERT ON author 16 FOR EACH ROW 17 BEGIN 18 DECLARE error_msg VARCHAR(255); 19 20 -- 检查author_id是否为空或格式不正确 21 IF NEW.author_id IS NULL OR NEW.author_id = '' THEN 22 SET error_msg = '作者ID不能为空'; 23 SIGNAL SQLSTATE '45000' SET MESSAGE_TEXT = error_msg; 24 END IF; 25 26 -- 检查author_id长度 27 IF LENGTH(NEW.author_id) > 50 THEN 28 SET error_msg = '作者ID长度不能超过50个字符'; 29 SIGNAL SQLSTATE '45000' SET MESSAGE_TEXT = error_msg; 30 END IF; 31 32 -- 检查authorname是否为空 33 IF NEW.authorname IS NULL OR NEW.authorname = '' THEN 34 SET error_msg = '作者姓名不能为空'; 35 SIGNAL SQLSTATE '45000' SET MESSAGE_TEXT = error_msg; 36 END IF; 37 38 -- 检查密码是否为空 39 IF NEW.password IS NULL OR NEW.password = '' THEN 40 SET error_msg = '密码不能为空'; 41 SIGNAL SQLSTATE '45000' SET MESSAGE_TEXT = error_msg; 42 END IF; </pre>

	<pre>-- 检查密码长度] IF LENGTH(NEW.password) < 6 THEN SET error_msg = '密码长度不能少于6个字符'; SIGNAL SQLSTATE '45000' SET MESSAGE_TEXT = error_msg; - END IF; -- 确保数值字段的默认值正确] IF NEW.writewords IS NULL THEN SET NEW.writewords = 0; - END IF;] IF NEW.worknum IS NULL THEN SET NEW.worknum = 0; - END IF;] IF NEW.level IS NULL OR NEW.level = '' THEN SET NEW.level = '普通'; - END IF; -END// -- 创建作者添加后的触发器（记录日志） CREATE TRIGGER after_author_insert AFTER INSERT ON author FOR EACH ROW BEGIN INSERT INTO author_log (author_id, action_type, details) VALUES (NEW.author_id, 'INSERT', CONCAT('成功添加作者: ID=', NEW.author_id, ', 姓名=', NEW.authername, ', 等级=', NEW.level)); - END// -- 恢复分隔符 DELIMITER ;</pre>
程序演示 (4分)	<p>说明：不违背触发器能够执行插入操作。</p> <div><h2>作者管理</h2><div><input type="text" value="请输入作者ID"/></div><div><div>🔍 搜索作者</div><div>➕ 添加作者</div></div><div><div><input type="text" value="1314520"/></div><div><input type="text" value="我是一条大锦鲤"/></div><div><input type="password" value="....."/></div><div><input type="text" value="1000"/></div><div><input type="text" value="1"/></div></div><div><div>localhost:5173</div><div>添加成功!</div><div>确定</div></div></div>

说明：违背触发器要求，不能够执行插入操作，系统报错。

程序演
示
(4分)

作者管理

🔍 搜索作者

+ 添加作者

1

⬆ ⬇ ⬆

11

⬆ ⬇ ⬆

localhost:5173

添加失败

确定

备注

6. 存储过程控制下的更新操作（18 分）

说明	<p>（1 分）简要说明该操作所要完成的功能；</p> <p>（1 分）简要说明该存储过程所要完成的功能；</p> <p>（2 分）说明该操作涉及操作的表（必须包含两张或两张以上的关系表，以“表名形式”描述）</p> <p>（1 分）表连接涉及字段描述（描述方式为“表 1. 属性=表 2. 属性”）</p> <p>（2 分）该操作会修改字段（以“表名. 字段名”的形式给出），以及修改规则，如新数值的计算方法、在何种条件下予以修改等；</p> <p>（6 分）实现该操作的关键代码（高级语言、SQL），截图即可；</p> <p>（5 分）如何执行该操作，按所述方法能够正常演示程序则给分。</p>	
功能描述 （1 分）	在一个小说管理系统中实现对作品信息的修改，当修改作品信息时，同时更改其他相关表中的外键。	
存储过程功能描述 （1 分）	当 author 中的信息进行更改时，比如更改了作者的笔名，那么 recommend（推荐榜）中的 authername 就需要进行修改，同时 reward（打赏表）表和 works（作品表）中的 authername 也需要进行更新。注意，作者的 ID 是无法进行更改的。	
涉及的关系表 （2 分）	<ul style="list-style-type: none"> • author • reward • recommend • works 	
表连接涉及字段 （1 分）	reward.author_id=author.author_id recommend.author_id=author.author_id	
更改字段 （2 分）	字段	规则
	author_id（作者 ID）	作者 ID 是作者的唯一标识符，无法进行更改
	password（作者账号密码）	新的作者账号密码替换旧的作者账号密码
	writewords（写了多少字）	新的码字字数替换旧的码字字数
	worknum（目前有多少作品）	新的作品个数替换旧的作品个数
	authername（作者笔名）	新的作者笔名替换旧的作者笔名

更新
代码
(3
分)

```
@app.route("/api/author/<author_id>", methods=["PUT"])
def update_author(author_id):
    try:
        data = request.get_json()

        # 只有当请求体中包含author_id且与URL中的不同时，才返回错误
        if "author_id" in data and str(data["author_id"]) != str(author_id):
            return jsonify({"error": "不可更改id"}), 400

        pen_name = data.get("authorname")
        password = data.get("password")
        writewords = data.get("writewords", 0)
        worknum = data.get("worknum", 0)
        level = data.get("level", "普通")

        cursor = db.cursor()


        # 调用存储过程
        cursor.callproc('UpdateAuthorInfo', [
            author_id, pen_name, password, writewords, worknum, level, 0, ''
        ])

        # 获取输出参数
        cursor.execute("SELECT @_UpdateAuthorInfo_6, @_UpdateAuthorInfo_7")
        result = cursor.fetchone()
        result_code = result[0] if result else 0
        message = result[1] if result else "未知错误"

        db.commit()
        cursor.close()

        if result_code == 1:
            return jsonify({"message": message}), 200
        elif result_code == 0:
            return jsonify({"error": "作者不存在"}), 404
        else:
            return jsonify({"error": message}), 500

    except Exception as e:
        return jsonify({"error": f"更新作者信息时发生错误: {str(e)}"}), 500
```

<p>创建 存储 过程 源码 (3 分)</p>	<pre> -- 创建更新作者信息的存储过程 DELIMITER // CREATE PROCEDURE UpdateAuthorInfo(IN p_author_id VARCHAR(50), IN p_authurname VARCHAR(100), IN p_password VARCHAR(255), IN p_writewords INT, IN p_worknum INT, IN p_level VARCHAR(20), OUT p_result_code INT, OUT p_message VARCHAR(255)) BEGIN DECLARE author_exists INT DEFAULT 0; DECLARE EXIT HANDLER FOR SQLEXCEPTION BEGIN ROLLBACK; SET p_result_code = -1; SET p_message = '更新作者信息时发生数据库错误'; END; START TRANSACTION; -- 检查作者是否存在 SELECT COUNT(*) INTO author_exists FROM author WHERE author_id = p_author_id; IF author_exists = 0 THEN SET p_result_code = 0; SET p_message = '作者不存在'; ROLLBACK; ELSE -- 更新作者信息 UPDATE author SET authurname = p_authurname, password = p_password, writewords = p_writewords, worknum = p_worknum, level = p_level WHERE author id = p author id; SET p_result_code = 1; SET p_message = '作者信息更新成功'; COMMIT; END IF; END // DELIMITER ; </pre>
<p>存储 过程 执行 源码 (1 分)</p>	<p>(截屏)</p>  <pre> # 调用存储过程 cursor.callproc('UpdateAuthorInfo', [author_id, pen_name, password, writewords, worknum, level, 0, '']) </pre>

程序
演示
(2
分)

说明：不违背存储过程，能够执行更新操作

管理员

作者日志退出

作者管理

2311366

搜索作者添加作者

作者ID

笔名

密码

写了多少字

目前有多少作品

作者ID: 2311366

笔名: 莫涵121

账号密码: 123456

写了多少字: 1111

作品数量: 0

等级: 黄金

修改作者信息

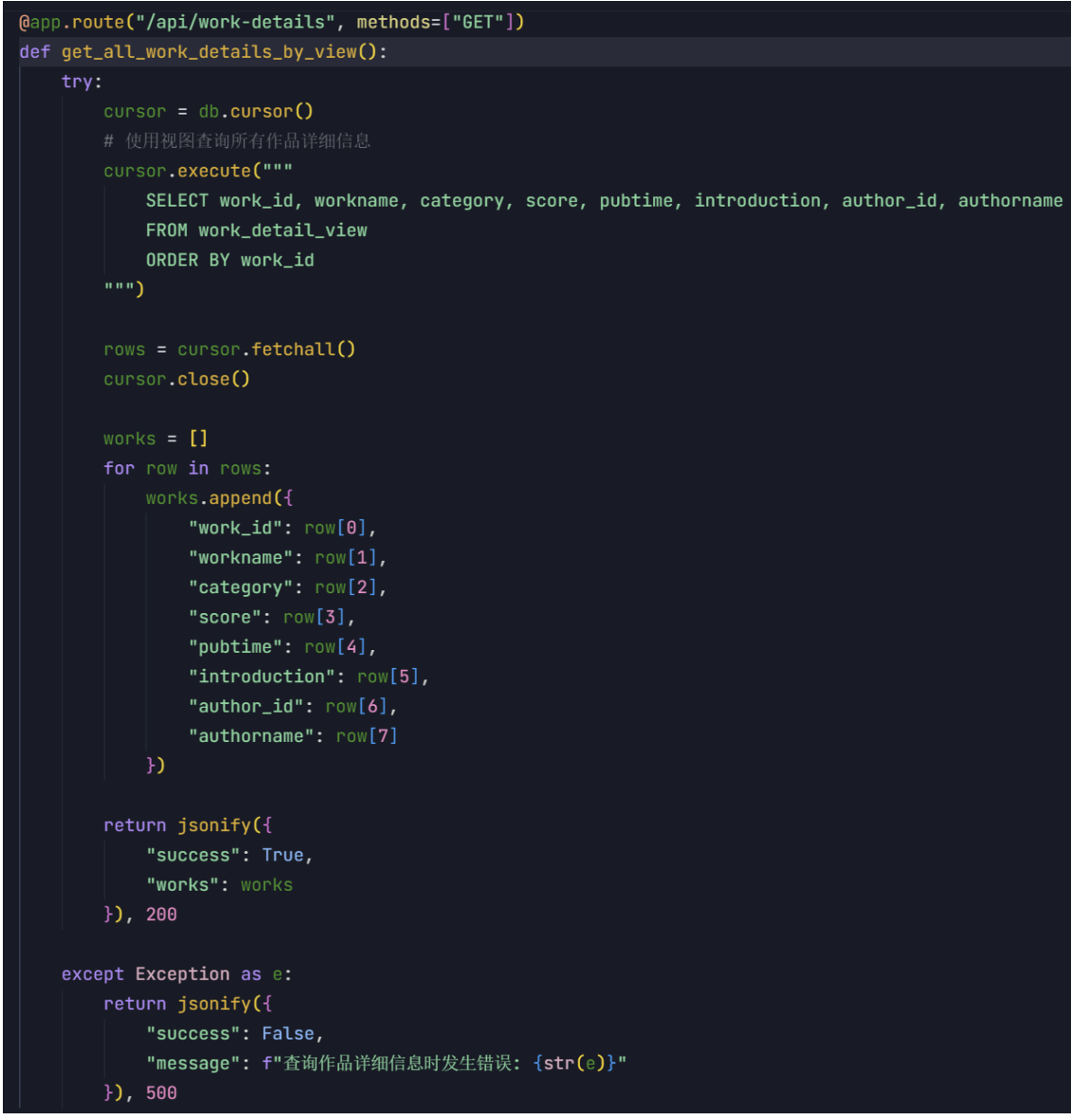
删除作者

	<div><div>修改作者信息</div><div><div>作者ID</div><div>2311366</div></div><div><div>莫涵是我呀</div><div>●●●●●</div></div><div><div>1111</div></div><div><div>1</div></div><div><div>黄金</div></div><div><div><input checked="" type="checkbox"/> 确认更新</div><div><input type="checkbox"/> 取消</div></div></div> <div><div>localhost:5173</div><div>更新成功! 作者信息更新成功</div><div>确定</div></div>
程序演示 (2分)	<p>说明：违背存储过程，系统报错：</p> <div><div>管理员</div><div>作者日志退出</div><div>作者管理</div><div><div>2311366</div><div>搜索作者添加作者</div></div><div><div>作者ID</div><div>笔名</div><div>密码</div><div>写了多少字</div><div>目前有多少作品</div></div><div><div>作者ID: 2311366</div><div>笔名: 莫涵121</div><div>账号密码: 123456</div><div>写了多少字: 1111</div><div>作品数量: 0</div><div>等级: 黄金</div></div><div><div>修改作者信息</div><div>删除作者</div></div></div>

	<div><div>修改作者信息</div><div><div>作者ID</div><div>2311366123</div></div><div>莫涵121</div><div>●●●●●</div><div>1111</div><div>1</div><div>黄金</div><div><div>✓ 确认更新</div><div>✕ 取消</div></div></div> <div><div>作者ID</div><div>2311366123</div><div>莫涵121</div><div>●●●●●</div><div>1111</div><div>1</div><div>黄金</div><div>localhost:5173</div><div>更新失败: 不可更改id</div><div>确定</div></div>
备注	

7. 含有视图的查询操作（15 分）

说明	(1 分) 简要说明该操作所要完成的功能; (1 分) 简要说明建立的该视图的功能; (2 分) 简要说明该操作涉及的关系数据表 (以 “表名” 的形式给出) (1 分) 简要说明表连接涉及的字段 (以 “表 1. 属性=表 2. 属性”) (6 分) 实现该操作的关键代码 (高级语言、SQL), 截图即可; (4 分) 如何执行该操作, 按所述方法能够正常演示程序则给分。
操作功能描述 (1 分)	该操作通过创建一个视图来简化复杂查询, 使得作品信息查询更加高效和易于管理。该视图将综合展示作品的详细信息, 包括作品的名字、作品的类型、作品的评分、作品的上架时间、作品的简介, 作者的 id 和名字。
视图功能描述 (1 分)	视图综合了作品表 (works) 和作者表 (author) 的信息, 提供一部作品的当前的相关信息, 同时还要给出作者表中的相关信息, 比如作者的名字和作者的 ID。
涉及的关系表 (2 分)	<ul style="list-style-type: none">• works• author
表连接字段 (1 分)	works.author_id=author.author_id works.authername=author.authername
创建视图代	(截屏)

码 (3 分)	<pre>-- 创建作品详细信息视图 CREATE VIEW work_detail_view AS SELECT w.work_id, w.workname, w.category, w.score, w.pubtime, w.introduction, w.author_id, w.authorname FROM works w;</pre>
查 询 代 码 (3 分)	<p>(截屏)</p>  <pre>@app.route("/api/work-details", methods=["GET"]) def get_all_work_details_by_view(): try: cursor = db.cursor() # 使用视图查询所有作品详细信息 cursor.execute(""" SELECT work_id, workname, category, score, pubtime, introduction, author_id, authorname FROM work_detail_view ORDER BY work_id """) rows = cursor.fetchall() cursor.close() works = [] for row in rows: works.append({ "work_id": row[0], "workname": row[1], "category": row[2], "score": row[3], "pubtime": row[4], "introduction": row[5], "author_id": row[6], "authorname": row[7] }) return jsonify({ "success": True, "works": works }), 200 except Exception as e: return jsonify({ "success": False, "message": f"查询作品详细信息时发生错误: {str(e)}" }), 500</pre>

程序演示 (4分)	<div data-bbox="331 203 1331 1507"><div data-bbox="335 208 1327 277"><div data-bbox="355 219 395 264"></div><div data-bbox="408 228 470 253">管理员</div><div data-bbox="1150 232 1227 250">作品日志</div><div data-bbox="1256 232 1302 250">退出</div></div><div data-bbox="758 344 904 387">作品管理</div><div data-bbox="557 452 569 472">1</div><div data-bbox="373 510 1291 562"><div data-bbox="373 510 667 562">搜索作品</div><div data-bbox="687 510 978 562">查询作品详情</div><div data-bbox="999 510 1291 562">+ 添加作品</div></div><div data-bbox="533 582 1131 1124"><div data-bbox="564 607 1106 640">作品ID</div><div data-bbox="564 667 1106 698">作品名称</div><div data-bbox="564 723 1106 754">类型</div><div data-bbox="564 781 1106 813">评分</div><div data-bbox="564 837 1106 869">作者ID</div><div data-bbox="564 893 1106 925">作者名称</div><div data-bbox="564 949 1106 981">yyyy / mm / dd</div><div data-bbox="564 1005 1106 1099">简介</div></div><div data-bbox="394 1144 1272 1485"><div data-bbox="422 1196 654 1225"> 作品详细信息视图</div><div data-bbox="435 1274 1174 1426"><div data-bbox="435 1274 537 1294">作者名：莫涵</div><div data-bbox="703 1274 829 1294">作者ID：2311366</div><div data-bbox="971 1274 1118 1294">作品名：种田我爽了</div><div data-bbox="435 1337 553 1357">作品类型：休闲</div><div data-bbox="703 1337 809 1357">作品评分：9.5</div><div data-bbox="971 1337 1174 1375">发布时间：Mon, 02 Jun 2025 20:12:19 GMT</div><div data-bbox="435 1402 584 1422">简介：种田使我快乐</div></div></div><div data-bbox="370 1525 1303 1888"><div data-bbox="399 1579 646 1608"> 作品详细信息视图</div><div data-bbox="413 1664 1197 1823"><div data-bbox="413 1664 518 1684">作者名：莫涵</div><div data-bbox="700 1664 831 1684">作者ID：2311366</div><div data-bbox="986 1664 1137 1684">作品名：种田我爽了</div><div data-bbox="413 1729 536 1749">作品类型：休闲</div><div data-bbox="700 1729 809 1749">作品评分：9.5</div><div data-bbox="986 1729 1197 1767">发布时间：Mon, 02 Jun 2025 20:12:19 GMT</div><div data-bbox="413 1798 569 1818">简介：种田使我快乐</div></div></div></div> <tr><td data-bbox="175 1895 260 1980">备注</td><td data-bbox="260 1895 1420 1980"></td></tr>	备注	
备注			