首先为了解决用户id无法自增的过程，进行了如下操作：

-- 获取当前的最大用户ID

DECLARE @maxID INT;

SELECT @maxID = ISNULL(MAX(用户ID), 0) FROM 用户;

-- 创建 SEQUENCE，起始值为最大用户ID的下一个数值

DECLARE @sequenceSQL NVARCHAR(MAX);

SET @sequenceSQL = 'CREATE SEQUENCE 用户ID序列 START WITH ' + CAST(@maxID + 1 AS NVARCHAR(10)) + ' INCREMENT BY 1;';

EXEC sp\_executesql @sequenceSQL;

CREATE TRIGGER trg\_用户ID自增

ON 用户

INSTEAD OF INSERT

AS

BEGIN

INSERT INTO 用户 (用户ID, 用户名, 创建时间, 角色ID, 邮箱, 手机号, 姓名, 密码)

SELECT NEXT VALUE FOR 用户ID序列, 用户名, 创建时间, 角色ID, 邮箱, 手机号, 姓名, 密码

FROM inserted;

END;

然后又禁用了

DISABLE TRIGGER trg\_用户ID自增 ON 用户;

CREATE PROCEDURE sp\_用户注册

@用户名 NVARCHAR(20),

@角色ID INT,

@邮箱 NVARCHAR(50),

@手机号 NVARCHAR(50),

@姓名 NVARCHAR(20),

@密码 NVARCHAR(20)

AS

BEGIN

SET NOCOUNT ON;

DECLARE @用户ID INT;

-- 插入用户信息到用户表

INSERT INTO 用户 (用户名, 创建时间, 角色ID, 邮箱, 手机号, 姓名, 密码)

VALUES (@用户名, GETDATE(), @角色ID, @邮箱, @手机号, @姓名, @密码);

-- 获取新插入用户的用户ID

SET @用户ID = SCOPE\_IDENTITY();

-- 插入角色和用户的关系到作为表

INSERT INTO 作为 (用户ID, 角色ID)

VALUES (@用户ID, @角色ID);

END;

**2.2.1 考生**

考生根据复试表填报个人报考信息和导师志愿。

考生查看复试录取结果。

CREATE PROCEDURE sp\_考生业务处理

@考生ID INT, -- 考生ID

@业务类型 VARCHAR(50), -- 业务类型: '填报志愿' 或 '查看录取结果'

@志愿ID INT,

@导师ID INT = NULL, -- 志愿导师ID (仅在填报志愿时使用)

@学科ID VARCHAR(50) = NULL, -- 所报学科ID (仅在填报志愿时使用)

@志愿顺序 INT = NULL, -- 志愿顺序 (仅在填报志愿时使用)

@是否接受方向调整 BIT = NULL -- 是否接受方向调整 (仅在填报志愿时使用)

AS

BEGIN

-- 判断业务类型

IF @业务类型 = '填报志愿'

BEGIN

-- 检查考生是否已提交该志愿信息，避免重复提交

IF EXISTS (SELECT 1 FROM 复试志愿 WHERE 考生ID = @考生ID AND 导师ID = @导师ID)

BEGIN

PRINT '已提交志愿信息，无法重复填报';

RETURN;

END

-- 插入考生志愿信息

-- 插入考生志愿信息

INSERT INTO 复试志愿 (志愿ID, 导师ID, 考生ID,学科ID, 志愿顺序, 是否接受方向调整, 创建时间, 更新时间)

VALUES (@志愿ID,@导师ID, @考生ID, @学科ID, @志愿顺序, @是否接受方向调整, CURRENT\_TIMESTAMP, CURRENT\_TIMESTAMP);

PRINT '志愿信息已成功提交';

END

ELSE IF @业务类型 = '查看录取结果'

BEGIN

-- 查询考生的复试和录取结果

SELECT

考生.考生ID AS 考生ID,

复试志愿.导师ID,

导师.导师ID AS 导师ID,

复试志愿.学科ID,

复试志愿.志愿顺序,

复试结果.录取状态,

复试结果.综合评价,

复试结果.更新时间 AS 录取更新时间

FROM 考生

LEFT JOIN 复试志愿 ON 考生.考生ID = 复试志愿.考生ID

LEFT JOIN 导师 ON 复试志愿.导师ID = 导师.导师ID

LEFT JOIN 复试结果 ON 考生.考生ID = 复试结果.考生ID

WHERE 考生.考生ID = @考生ID;

PRINT '录取结果已显示';

END

ELSE

BEGIN

-- 错误处理：无效的业务类型

PRINT '无效的业务类型，请指定 "填报志愿" 或 "查看录取结果"';

END

END;

测试：

EXEC sp\_考生业务处理

@考生ID = 11,

@业务类型 = '填报志愿',

@导师ID = 10,

@学科ID = '083200-04',

@志愿顺序 = 1,

@志愿ID=102,

@是否接受方向调整 = 1;

EXEC sp\_考生业务处理

@考生ID = 11,

@志愿ID=102,

@业务类型 = '查看录取结果';



在实现触发器（保证数据一致性）的创建以后测试：

EXEC dbo.sp\_考生业务处理

@考生ID = 101,

@业务类型 = '填报志愿',

@志愿ID = 1,

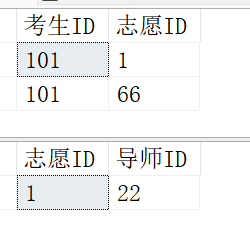
@导师ID = 22,

@学科ID = '070503-01',

@志愿顺序 = 1,

@是否接受方向调整 = 1;

填报志愿时，填报与选择一同更新：



2.2.2

CREATE PROCEDURE sp\_导师操作

@操作类型 NVARCHAR(20), -- 操作类型，'更新简介' 或 '确认学生' 或 '查看选择结果'

@导师ID INT, -- 导师ID

@简介 VARCHAR(1000) = NULL, -- 导师简介，更新简介时使用

@考生ID INT = NULL -- 考生ID，确认选择时使用

AS

BEGIN

-- 判断操作类型

IF @操作类型 = '更新简介'

BEGIN

-- 更新导师简介

UPDATE 导师

SET 简介 = @简介,

更新时间 = GETDATE()

WHERE 导师ID = @导师ID;

PRINT '导师简介已更新';

END

ELSE IF @操作类型 = '确认学生'

BEGIN

-- 确认选择的学生，将复试志愿的志愿ID插入到选择表中

IF EXISTS (SELECT 1 FROM 复试志愿 WHERE 导师ID = @导师ID AND 考生ID = @考生ID)

BEGIN

INSERT INTO 选择 (志愿ID, 导师ID)

SELECT 志愿ID, 导师ID

FROM 复试志愿

WHERE 导师ID = @导师ID AND 考生ID = @考生ID;

PRINT '学生已确认选择';

END

ELSE

BEGIN

PRINT '该学生未选择此导师为志愿';

END

END

ELSE IF @操作类型 = '查看选择结果'

BEGIN

-- 查看导师的所有确认选择结果，包括考生信息和录取状态

SELECT r.志愿ID,

c.考生ID,

c.性别,

c.生源地,

r.志愿顺序,

fr.录取状态,

fr.综合评价

FROM 复试志愿 r

JOIN 考生 c ON r.考生ID = c.考生ID

LEFT JOIN 复试结果 fr ON fr.考生ID = c.考生ID AND fr.最终导师ID = @导师ID

WHERE r.导师ID = @导师ID;

END

ELSE

BEGIN

PRINT '无效的操作类型';

END

END;

测试：

EXEC sp\_导师操作

@操作类型 = '更新简介',

@导师ID = 1,

@简介 = '从事人工智能与机器学习研究';

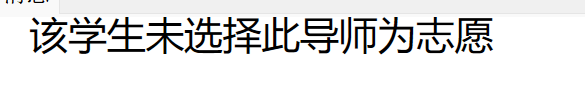


EXEC sp\_导师操作

@操作类型 = '确认学生',

@导师ID = 3,

@考生ID = 1;



EXEC sp\_导师操作

@操作类型 = '查看选择结果',

@导师ID = 1;



**2.2.3 学科秘书**

推动导师学生志愿选择过程。

CREATE PROCEDURE sp\_推动导师学生志愿选择过程

@用户ID INT, -- 当前用户ID，输入参数

@复试结果ID INT -- 复试结果ID，指定需要更新的复试结果

AS

BEGIN

-- 检查用户是否为学科秘书

DECLARE @角色名称 VARCHAR(20);

-- 获取当前用户的角色名称

SELECT @角色名称 = 角色名称

FROM 作为

JOIN 角色 ON 作为.角色ID = 角色.角色ID

WHERE 作为.用户ID = @用户ID;

-- 如果角色是学科秘书，则执行更新操作

IF @角色名称 = '学科秘书'

BEGIN

-- 更新复试结果状态为待确认

UPDATE 复试结果

SET 待确认=1, -- 将录取状态从其他状态改为待确认

更新时间 = CURRENT\_TIMESTAMP -- 更新时间为当前时间

WHERE 复试结果ID = @复试结果ID

AND 待确认 IS NULL; -- 只更新状态为 NULL 的记录

END

ELSE

BEGIN

-- 如果不是学科秘书，返回错误信息

PRINT '权限不足，只有学科秘书可以执行此操作';

END

END;

GO

测试：

EXEC sp\_推动导师学生志愿选择过程 @用户ID = 5, @复试结果ID = 1;



**2.2.4 学科负责人**

确定学科研究方向。

CREATE PROCEDURE sp\_确定学科研究方向

@用户ID INT, -- 当前用户ID，输入参数

@学科ID VARCHAR(15), -- 学科ID，输入参数

@研究方向名称 VARCHAR(100), -- 新的研究方向名称，输入参数

@描述 VARCHAR(500) -- 研究方向的描述，输入参数

AS

BEGIN

-- 声明角色名称变量

DECLARE @角色名称 VARCHAR(20);

-- 获取当前用户的角色名称

SELECT @角色名称 = 角色名称

FROM 作为

JOIN 角色 ON 作为.角色ID = 角色.角色ID

WHERE 作为.用户ID = @用户ID;

-- 判断当前用户是否是学科负责人

IF @角色名称 = '学科负责人'

BEGIN

-- 如果是学科负责人，则进行学科研究方向的更新操作

IF EXISTS (SELECT 1 FROM 学科 WHERE 学科ID = @学科ID)

BEGIN

-- 如果该学科存在，更新学科研究方向

UPDATE 学科

SET 学科研究方向 = @研究方向名称,

学科概述 = @描述, -- 也可以更新学科概述或其他字段

更新时间 = CURRENT\_TIMESTAMP

WHERE 学科ID = @学科ID;

END

ELSE

BEGIN

-- 如果该学科ID不存在，则返回错误信息

PRINT '该学科ID不存在，请检查学科ID';

END

END

ELSE

BEGIN

-- 如果不是学科负责人，则返回权限不足的提示

PRINT '权限不足，只有学科负责人可以执行此操作';

END

END;

测试：

EXEC sp\_确定学科研究方向

@用户ID = 4,

@学科ID = '070503-01',

@研究方向名称 = '智能制造',

@描述 = '研究智能制造相关技术';



**2.2.5 研究生管理秘书**

录入考生初试信息。

录入考生复试信息。

收集和审核复试考生信息。

发布录取公示结果。

CREATE PROCEDURE sp\_管理考生信息

@用户ID INT, -- 当前用户ID

@考生ID INT, -- 考生ID

@考试地点 VARCHAR(50),

@科目名称 VARCHAR(50),

@考试科目ID INT, -- 考试科目ID（用于录入初试信息）

@初试成绩 DECIMAL(10, 2), -- 初试成绩

@复试成绩 DECIMAL(10, 2), -- 复试成绩

@复试志愿ID INT,

@导师ID INT, -- 导师ID（用于录入复试信息）

@学科ID VARCHAR(50), -- 学科ID（用于录入复试信息）

@志愿顺序 INT, -- 复试志愿顺序

@是否接受方向调整 BIT, -- 是否接受方向调整

@复试结果ID INT = NULL, -- 复试结果ID（可选，若有复试结果，则进行审核和发布）

@审核状态 VARCHAR(50) = '待确认', -- 默认审核状态，若复试结果已确认则可更新

@业务类型 VARCHAR(50) -- 业务类型：'录入初试信息' 或 '录入复试信息'

AS

BEGIN

-- 检查用户是否为研究生管理秘书

DECLARE @角色名称 VARCHAR(20);

-- 获取当前用户的角色名称

SELECT @角色名称 = 角色名称

FROM 作为

JOIN 角色 ON 作为.角色ID = 角色.角色ID

WHERE 作为.用户ID = @用户ID;

-- 如果角色是研究生管理秘书，则执行操作

IF @角色名称 = '研究生管理秘书'

BEGIN

-- 根据业务类型判断执行不同的操作

IF @业务类型 = '录入初试信息'

BEGIN

-- 1. 录入考生初试信息

-- 2. 更新考试科目的类别为 '初试'

INSERT INTO 考试科目 (考试科目ID, 类别, 名称, 考试时间, 考试地点)

VALUES (@考试科目ID, '初试', @科目名称, CURRENT\_TIMESTAMP, @考试地点);

INSERT INTO 成绩 (考生ID, 考试科目ID, 数值)

VALUES (@考生ID, @考试科目ID, @初试成绩);

PRINT '初试信息录入成功';

END

ELSE IF @业务类型 = '录入复试信息'

BEGIN

-- 1. 录入考生复试信息

-- 2. 更新考试科目的类别为 '初试'

INSERT INTO 考试科目 (考试科目ID, 类别, 名称, 考试时间, 考试地点)

VALUES (@考试科目ID, '复试', @科目名称, CURRENT\_TIMESTAMP, @考试地点);

INSERT INTO 成绩 (考生ID, 考试科目ID, 数值)

VALUES (@考生ID, @考试科目ID, @复试成绩);

INSERT INTO 复试志愿 (志愿ID考生ID, 导师ID, 学科ID, 志愿顺序, 是否接受方向调整, 更新时间, 创建时间)

VALUES (复试志愿ID,@考生ID, @导师ID, @学科ID, @志愿顺序, @是否接受方向调整, CURRENT\_TIMESTAMP, CURRENT\_TIMESTAMP);

PRINT '复试信息录入成功';

END

-- 3. 收集并审核复试信息（如果复试结果ID不为NULL，进行审核）

IF @复试结果ID IS NOT NULL

BEGIN

-- 获取并审核复试结果

UPDATE 复试结果

SET 录取状态 = @审核状态,

更新时间 = CURRENT\_TIMESTAMP

WHERE 复试结果ID = @复试结果ID

AND 录取状态 = '待确认';

END;

-- 4. 发布录取公示结果（如果复试结果已确认）

IF @审核状态 = '已确认'

BEGIN

UPDATE 复试结果

SET 录取状态 = '已录取',

更新时间 = CURRENT\_TIMESTAMP

WHERE 复试结果ID = @复试结果ID

AND 录取状态 = '待审核';

-- 可选择向外部系统或相关人员发布录取信息

PRINT '录取公示已发布';

END;

PRINT '操作已成功完成';

END

ELSE

BEGIN

-- 如果用户角色不是研究生管理秘书，返回错误信息

PRINT '权限不足，只有研究生管理秘书可以执行此操作';

END

END;

测试：

EXEC sp\_管理考生信息

@用户ID = 7, -- 当前用户ID

@考生ID = 6, -- 考生ID

@考试地点 = '考场A', -- 考试地点

@科目名称 = '线性代数', -- 科目名称

@考试科目ID = 688, -- 考试科目ID

@初试成绩 = 85.5, -- 初试成绩

@复试成绩 = NULL, -- 复试成绩为空

@导师ID = NULL, -- 复试信息不涉及导师

@学科ID = NULL, -- 复试信息不涉及学科

@志愿顺序 = NULL, -- 复试信息不涉及志愿顺序

@是否接受方向调整 = NULL, -- 复试信息不涉及方向调整

@复试结果ID = NULL, -- 复试结果为空

@审核状态 = '待确认', -- 复试结果审核状态

@业务类型 = '录入初试信息'; -- 业务类型：录入初试信息



EXEC sp\_管理考生信息

@用户ID = 7, -- 当前用户ID

@考生ID = 8, -- 考生ID

@考试地点 = '考场B', -- 考试地点

@科目名称 = '面试', -- 科目名称

@考试科目ID = 112, -- 考试科目ID

@初试成绩 = NULL, -- 初试成绩为空

@复试成绩 = 88.0, -- 复试成绩

@导师ID = 12, -- 导师ID

@复试志愿ID=30,

@学科ID = '070503-01', -- 学科ID

@志愿顺序 = 3, -- 志愿顺序

@是否接受方向调整 = 1, -- 是否接受方向调整

@复试结果ID = 53, -- 复试结果为空

@审核状态 = '待确认', -- 复试结果审核状态

@业务类型 = '录入复试信息'; -- 业务类型：录入复试信息



EXEC sp\_管理考生信息

@用户ID = 2, -- 当前用户ID（用户2不是研究生管理秘书）

@考生ID = 1004, -- 考生ID

@考试地点 = '考场D', -- 考试地点

@科目名称 = '计算机基础', -- 科目名称

@考试科目ID = 504, -- 考试科目ID

@初试成绩 = 78.0, -- 初试成绩

@复试成绩 = NULL, -- 复试成绩为空

@导师ID = NULL, -- 复试信息不涉及导师

@学科ID = NULL, -- 复试信息不涉及学科

@志愿顺序 = NULL, -- 复试信息不涉及志愿顺序

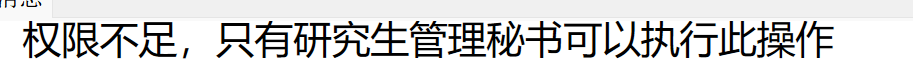
@是否接受方向调整 = NULL, -- 复试信息不涉及方向调整

@复试结果ID = NULL, -- 复试结果为空

@审核状态 = '待确认', -- 复试结果审核状态

@业务类型 = '录入初试信息', -- 业务类型：录入初试信息

@复试志愿ID=0;



**2.2.6 研究生主管领导**

审议考生和学生选择结果。

CREATE PROCEDURE sp\_审议考生选择结果

@主管领导ID INT, -- 当前用户ID，确保具有主管领导权限

@考生ID INT, -- 考生ID

@复试结果ID INT, -- 复试结果ID，用于指定审核记录

@审核状态 VARCHAR(50), -- 审核状态，例如 '已确认' 或 '未通过'

@审核意见 NVARCHAR(200) = NULL -- 审核意见，可选字段，提供审核说明

AS

BEGIN

-- 检查用户是否为研究生主管领导

DECLARE @角色名称 VARCHAR(20);

-- 获取当前用户的角色名称

SELECT @角色名称 = 角色名称

FROM 作为

JOIN 角色 ON 作为.角色ID = 角色.角色ID

WHERE 作为.用户ID = @主管领导ID;

-- 确认用户角色为研究生主管领导后执行操作

IF @角色名称 = '研究生主管领导'

BEGIN

-- 更新复试结果，设置审核状态和审核意见

UPDATE 复试结果

SET 录取状态 = @审核状态,

综合评价 = @审核意见,

更新时间 = CURRENT\_TIMESTAMP

WHERE 考生ID = @考生ID

AND 复试结果ID = @复试结果ID;

-- 若审核通过，则将最终状态设置为“已录取”，否则视情况修改为其他状态

IF @审核状态 = '已录取'

BEGIN

PRINT '复试审核已确认，录取状态已更新为已录取。';

END

ELSE

BEGIN

PRINT '复试审核未通过，已记录审核意见。';

END

END

ELSE

BEGIN

-- 如果用户角色不是研究生主管领导，返回错误信息

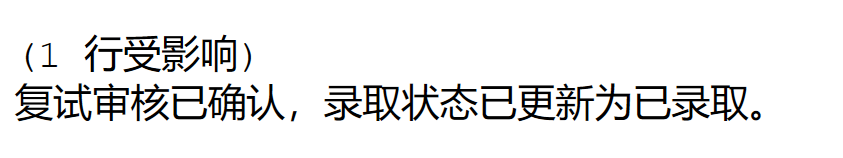
PRINT '权限不足，只有研究生主管领导可以执行此操作';

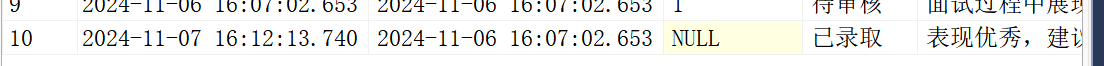
END

END;

测试：

EXEC sp\_审议考生选择结果 @主管领导ID = 10, @考生ID = 10, @复试结果ID = 10, @审核状态 = '已录取', @审核意见 = '表现优秀，建议录取';





**2.2.7 监督成员**

同研究生主管领导，但没有审核权，可查看全过程。

CREATE PROCEDURE sp\_查看考生选择全过程

@监督成员ID INT, -- 当前用户ID，确保具有监督成员权限

@考生ID INT -- 考生ID，用于查看特定考生的过程

AS

BEGIN

-- 检查用户是否为监督成员

DECLARE @角色名称 VARCHAR(20);

-- 获取当前用户的角色名称

SELECT @角色名称 = 角色名称

FROM 作为

JOIN 角色 ON 作为.角色ID = 角色.角色ID

WHERE 作为.用户ID = @监督成员ID;

-- 确认用户角色为监督成员后执行操作

IF @角色名称 = '监督成员'

BEGIN

-- 查询考生的复试和录取信息全过程

SELECT

考生.考生ID,

考生.考生类别,

复试志愿.导师ID,

复试志愿.志愿顺序,

复试志愿.是否接受方向调整,

复试结果.录取状态,

复试结果.综合评价,

复试结果.更新时间 AS 最后更新时间

FROM 考生

LEFT JOIN 复试志愿 ON 考生.考生ID = 复试志愿.考生ID

LEFT JOIN 导师 ON 复试志愿.导师ID = 导师.导师ID

LEFT JOIN 复试结果 ON 考生.考生ID = 复试结果.考生ID

WHERE 考生.考生ID = @考生ID;

PRINT '考生选择全过程已显示';

END

ELSE

BEGIN

-- 如果用户角色不是监督成员，返回错误信息

PRINT '权限不足，只有监督成员可以执行此操作';

END

END;

测试：

EXEC sp\_查看考生选择全过程 @监督成员ID = 6, @考生ID = 6;

