

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	2211290	姓名	姚知言	专业	计算机科学与技术
项目名称	Iwork 公司管理系统				
必备环境	Qt 6.8.0(MSVC2019)；MySQL Server 8.0.35；ODBC 8.0.33				
系统主要功能简介（4 分）	<p>在该管理系统中，运行后可以看到登录界面（见下图 1）。</p> <p>员工可以通过个人 employeeID 和 pwd 登录，通过数据库表进行比对，登录后可以查看个人信息（见下图 2）。</p> <p>管理员通过预留 admin 账号和密码登录，登录后可对数据库各表进行添加，更新，删除与查看（见下图 3）。</p> <p>主要分为员工表 employee 和部门表 department 的操作，员工表 employee 的两个子表的操作在员工表操作页面进行。</p> <p>对员工表（及子表）的查看分为根据 employeeID 直接搜索和根据 departmentID 分类筛选搜索（见下图 4）。</p> <p>对各表的增加，修改，删除操作均设置了灵活的对某一属性或几属性的操作以及设为空值的设定（除了不能被设为空值的特殊列）。</p> <p>对各表的更新操作也存在一些事务应用，触发器，存储过程以及视图的应用，以对系统进行优化。</p>				
系统主要页面截图（6 分）	<p>在该部分尽量少的对后续篇目有机会展示的视图进行展示，互为补充。</p> <p>图 1：登录界面</p> <p>该弹窗为点击问号处唤起。</p>				

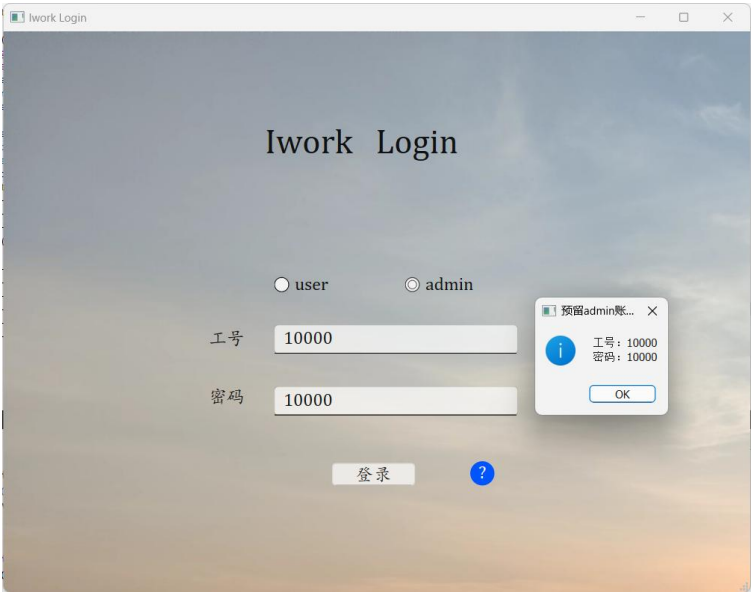


图 2：登录成功页面-员工



图 3：登录成功页面-管理员



图 4：员工信息查询-根据部门 ID 筛选

Iwork Admin

2024-05-23 00:47:25

Welcome: admin

部门ID9988-相声部

查询

返回上层

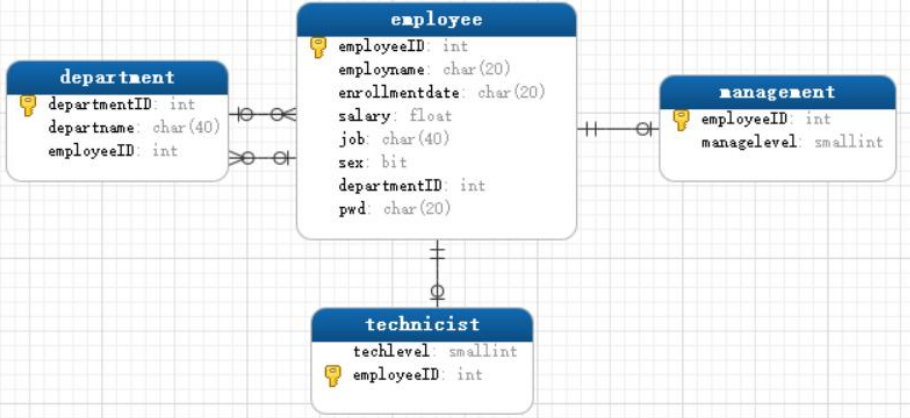
工号	姓名	性别	入职时间	职务	技术	管理	工资	密码
1 10008	Ear ly8				0	0		

此处举出第 7 部分展示的另一例子，与其互为补充。

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			
	高级语言	1. C++(with Qt)			
连接串 分析 (6 分)		序号	名称	功能说明	取值
		1	addDatabase	创建数据库连接，以 ODBC 作为数据库驱动	QODBC
		2	setHostName	设置数据库服务器的 IP 地址	127.0.0.1
		3	setPort	设置数据可服务器的端口	3306
		4	setDBname	设置连接的数据库名称，与 ODBC 保持一致	myodbc
		5	setUserName	设置连接数据库的用户名	root
		6	setPassword	设置连接数据库的密码	1433223aaa
		7	open	打开数据库	无
连接串代码 (截屏) (2 分)		<pre>QSqlDatabase db = QSqlDatabase::addDatabase("QODBC"); db.setHostName("127.0.0.1"); db.setPort(3306); db.setDatabaseName("myodbc"); db.setUserName("root"); db.setPassword("1433223aaa"); bool ok = db.open(); if(ok){ w.show(); } else{ QMessageBox::information(NULL,"失败","数据库连接失败!"); qDebug()<<"error open database because"<<db.lastError().text(); }</pre>			
备注		下方判断语句表示如果成功连接数据库则打开主界面，否则不打开并弹出报错信息。			

3. 数据库设计（14 分）

说明	<p>（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；</p> <p>（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。</p>				
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	employee	employeeID	departmentID	department.deparementID
	2	department	departmentID	employeeID	employee.employeeID
	3	technicist	employeeID	employeeID	employee.employeeID
	4	management	employeeID	employeeID	employee.employeeID
关系图 (4)					
备注	department.employeeID 存储的本质上是该部门的部长信息。				

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)	删除一个部门，并将部门中的所有员工转移至指定的新部门或 id 为 0 默认部门（若未给出）。
涉及的表 (2 分)	employee, department
表连接涉及字段 (1 分)	employee.departmentID=department.departmentID

	字段	规则
删除条件字段描述 (1 分)	department.departmentID department.departmentname department.employeeID	根据给定的 departmentID 将属性进行删除
	employee.departmentID	若未指定新 departmentID, 则将 departmentID 清除 (置为 0), 若指定了新的 departmentID, 则将其更新为新的 departmentID
代码 (4 分)	<pre> void depop::on_sc_clicked(){ bool c1; int val=ui->e4->text().toInt(&c1); if(!c1 !val){ QMessageBox::critical(NULL,"错误","输入的部门ID不合法, 移除失败! "); return; } int val2=0; if(!ui->e7->text().isEmpty()){ bool c2; val2=ui->e7->text().toInt(&c2); if(!c2){ QMessageBox::critical(NULL,"错误","输入的新部门ID不合法, 移除失败! "); return; } } QSqlQuery query_update; QSqlQuery query_delete; QSqlQuery transaction_start; QSqlQuery transaction_commit; QSqlQuery transaction_rollback; transaction_start.exec("START TRANSACTION"); query_update.prepare("update employee set departmentID =:value1 where departmentID =:value2"); query_update.bindValue(":value1", val2); query_update.bindValue(":value2", val); query_delete.prepare("DELETE FROM department WHERE departmentID = :value"); query_delete.bindValue(":value", val); bool ok1=query_update.exec(); bool ok2=query_delete.exec(); if(ok1 && ok2) { transaction_commit.exec("COMMIT"); QMessageBox::information(NULL,"操作完成","操作完成! "); }else { transaction_rollback.exec("ROLLBACK"); QMessageBox::critical(NULL,"错误","操作失败! "); } } </pre>	

程序演示(4分)

假设我们在一开始在表中存在以下数据。

```
mysql> select * from employee;
+-----+-----+-----+-----+-----+-----+-----+-----+
| employeeID | employname | enrollmentdate | salary | job        | sex | departmentID | pwd |
+-----+-----+-----+-----+-----+-----+-----+-----+
| 10001 | bob      | 0              | NULL   | 0          | 1   | 1001         | 10001 |
| 10003 | cathy    | 0              | NULL   | Sleeping   | 0   | 1001         | 10003 |
| 10004 | cat      | NULL           | NULL   | NULL       | 1   | 1002         | NULL  |
| 10008 | Early8   | NULL           | NULL   | NULL       | NULL | 1002         | NULL  |
| 10009 | Amy      | NULL           | NULL   | NULL       | 0   | 1003         | NULL  |
+-----+-----+-----+-----+-----+-----+-----+-----+
5 rows in set

mysql> select * from department;
+-----+-----+-----+
| departmentID | departname | employeeID |
+-----+-----+-----+
| 0 | NULL | NULL |
| 1001 | 武装部 | 10001 |
| 1002 | 文体部 | NULL |
| 1003 | 宣传部 | NULL |
+-----+-----+-----+
4 rows in set
```

若将这两个操作视为原子操作，以将 departmentID: 1003 迁移至 departmentID: 1004 为例。由于外键以来第一个更新语句未能成功，但由于 cascade 设置第二个删除语句成功，同时部门 1003 下的员工 Amy 也被删除。

```
mysql> update employee set departmentID =1004 where departmentID =1003
-> ;
1452 - Cannot add or update a child row: a foreign key constraint fails (`community`.`employee` (departmentID) ON DELETE CASCADE ON UPDATE CASCADE)
mysql> DELETE FROM department WHERE departmentID =1003;
Query OK, 1 row affected
```

结果如下:

```
mysql> select * from employee;
+-----+-----+-----+-----+-----+-----+-----+-----+
| employeeID | employname | enrollmentdate | salary | job        | sex | departmentID | pwd |
+-----+-----+-----+-----+-----+-----+-----+-----+
| 10001 | bob      | 0              | NULL   | 0          | 1   | 1001         | 10001 |
| 10003 | cathy    | 0              | NULL   | Sleeping   | 0   | 1001         | 10003 |
| 10004 | cat      | NULL           | NULL   | NULL       | 1   | 1002         | NULL  |
| 10008 | Early8   | NULL           | NULL   | NULL       | NULL | 1002         | NULL  |
+-----+-----+-----+-----+-----+-----+-----+-----+
4 rows in set

mysql> select * from department;
+-----+-----+-----+
| departmentID | departname | employeeID |
+-----+-----+-----+
| 0 | NULL | NULL |
| 1001 | 武装部 | 10001 |
| 1002 | 文体部 | NULL |
+-----+-----+-----+
3 rows in set
```

接下来我们尝试在前端完成该操作将 departmentID: 1002 迁移至 departmentID: 1004。理论上，若两条操作为原子操作，则结果将与刚才一样。

Iwork Admin

2024-05-21 23:48:57

Welcome: admin

部门添加

部门ID

部门名

部长ID

注：
1.除部门ID外，其他均非必填项。
2.部门ID不可与已有部门ID重复。
3.部长必须存在，且设置部长后，部长的部门自动调整至新部门。

提交

部门修改/移除

部门ID

部门名

部长ID

新部门ID

注：
1.部门移除操作必须填写部门ID，若填写新部门ID则其中员工迁移至新部门，否则迁移至默认（ID：0）部门。
2.部门修改操作是通过给定的部门ID修改部门名和部长，部门ID不可缺省，部门名和部长若仅需修改一处，则另一处无需填写，若需改为空值则勾选复选框，复选框代表高级干预输入。

☐ null

☐ null

修改 移除

返回

点击移除后，返回错误的提示窗。

Iwork Admin

2024-05-21 23:50:58

Welcome: admin

部门添加

部门ID

部门名

部长ID

注：
1.除部门ID外，其他均非必填项。
2.部门ID不可与已有部门ID重复。
3.部长必须存在，且设置部长后，部长的部门自动调整至新部门。

提交

部门修改/移除

部门ID

部门名

部长ID

新部门ID

注：
1.部门移除操作必须填写部门ID，若填写新部门ID则其中员工迁移至新部门，否则迁移至默认（ID：0）部门。
2.部门修改操作是通过给定的部门ID修改部门名和部长，部门ID不可缺省，部门名和部长若仅需修改一处，则另一处无需填写，若需改为空值则勾选复选框，复选框代表高级干预输入。

☐ null

☐ null

修改 移除

返回

为确认数据库没有失去连接，此时创建一个新部门作为对照。

Iwork Admin

2024-05-21 23:53:23

Welcome: admin

部门添加

部门ID

1005

部门名

舞蹈部

部长ID

提交

注:

1.除部门ID外,其他均非必填项。
2.部门ID不可与已有部门ID重复。
3.部长必须存在,且设置部长后,部长的部门自动调整至新部门。

操作完成

操作完成!

OK

部门修改/移除

部门ID

部门名

☐ null

部长ID

☐ null

新部门ID

修改

移除

注:

1.部门移除操作必须填写部门ID,若填写新部门ID则其中员工迁移至新部门,否则迁移至默认(ID: 0)部门。
2.部门修改操作是通过给定的部门ID修改部门名和部长,部门ID不可缺省,部门名和部长若仅需修改一处,则另一处无需填写,若需改为空值则必须填写,复选框传去略高于输入。

返回

此时结果如下，我们可以成功验证其在执行失败后正确完成回滚，很好的执行了事务应用，

```
mysql> select * from employee;
```

employeeID	employname	enrollmentdate	salary	job	sex	departmentID	pwd
10001	bob	0	NULL	0	1	1001	10001
10003	cathy	0	NULL	Sleeping	0	1001	10003
10004	cat	NULL	NULL	NULL	1	1002	NULL
10008	Early8	NULL	NULL	NULL	NULL	1002	NULL

4 rows in set

```
mysql> select * from department;
```

departmentID	departname	employeeID
0	NULL	NULL
1001	武装部	10001
1002	文体部	NULL
1005	舞蹈部	NULL

4 rows in set

再演示一下正常 commit 的结果，删除 departmentID: 1002 迁移至 departmentID: 1005。

	<div><div>Iwork Admin</div><div>2024-05-21 23:57:37</div><div>Welcome: admin</div></div> <div><div><div>部门添加</div><div><div>部门ID</div><div>部门名</div><div>部长ID</div></div><div><div>提交</div></div><div><div>注:</div><div>1.除部门ID外,其他均非必填项。</div><div>2.部门ID不可与已有部门ID重复。</div><div>3.部长必须存在,且设置部长后,部长的部门自动调整至新部门。</div></div></div><div><div><div>操作完成</div><div>操作完成!</div><div>OK</div></div></div><div><div><div>部门修改/移除</div><div><div>部门ID</div><div>部门名</div><div>部长ID</div><div>新部门ID</div></div><div><div>修改</div><div>移除</div></div><div><div>注:</div><div>1.部门移除操作必须填写部门ID,若填写新部门ID则其中员工迁移至新部门,否则迁移至默认(ID: 0)部门。</div><div>2.部门修改操作是通过给定的部门ID修改部门名和部长,部门ID不可缺省,部门名和部长若仅需修改一处,则另一处无需填写,若需改为空值则必须清洗并输入。</div></div></div><div><div>返回</div></div></div><div><p>此时的结果如下。成功完成了事务操作。</p><pre>mysql> select * from employee; +-----+-----+-----+-----+-----+-----+-----+-----+ employeeID employname enrollmentdate salary job sex departmentID pwd +-----+-----+-----+-----+-----+-----+-----+-----+ 10001 bob 0 NULL 0 1 1001 10001 10003 cathy 0 NULL Sleeping 0 1001 10003 10004 cat NULL NULL NULL 1 1005 NULL 10008 Early8 NULL NULL NULL NULL 1005 NULL +-----+-----+-----+-----+-----+-----+-----+-----+ 4 rows in set mysql> select * from department; +-----+-----+-----+ departmentID departname employeeID +-----+-----+-----+ 0 NULL NULL 1001 武装部 10001 1005 舞蹈部 NULL +-----+-----+-----+ 3 rows in set</pre></div></div>
备注	从 navicat 上的原子执行作为对比，可以验证出 rollback 操作的正常执行。以最后的例子验证出 commit 操作的正常执行。

5. 触发器控制下的添加操作（20 分）

说明	<div><div>(1 分) 简要说明该操作所要完成的功能；</div><div>(2 分) 简要说明该触发器所要完成的功能</div><div>(1 分) 该操作会涉及的表（以“表名”的形式给出）。</div><div>(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；</div><div>(6 分) 实现该操作的关键代码（高级语言、SQL），截图即可；</div><div>(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。</div></div>
功能描述 (1 分)	在此部分中，我们实现了对 department 表的添加操作。也就是在点击添加按钮的时候，首先检查添加是否合法，其次若合法且指定了 employeeID 作为部长，则将对对应 employee 表中的员工的 departmentID 更改为新的 departmentID。为了更好的完成这个操作，我们实现了两个触发器 checkdpid 和 updated。同时，为了适应多元化的

	环境，也以分路选择结构设置的各种情况下的 insert。	
触发器描述 (2 分)	<p>对于触发器 checkdpid，其功能为在插入 department 表之前检查输入的 departmentID 是否在 1001-9999 的范围内，由于其已经作为 department 表的主键，所以不用添加触发器也可以在取值重复的前提下弹出错误。若失败，则报出一个错误。</p> <p>对于触发器 updated，其功能为在这次添加成功后，若这次添加指定了 employeeID，则将 employee 表中对应 employeeID 的员工的 departmentID 更改为新的 departmentID。(因为如果进行指定了，根据外键约束必须存在这样一个 employeeID 否则无法插入成功)</p>	
涉及的表 (1 分)	employee, department	
输入数据 (2 分)	字段	规则
	department.departmentID	必须设置(主键)，范围在 1001-9999，不能重复
	department.departmentname	不必须设置，无限制
	department.employeeID	不必须设置，但若设置需要满足 employee.employeeID 的外键约束
插入操作 源码 (3 分)	<pre> void depop::on_sm_clicked() { bool c1; int val=ui->e1->text().toInt(&c1); if(!c1){ QMessageBox::critical(NULL,"错误","输入的部门ID不是数字，创建失败!"); return; } if(lui->e2->text().isEmpty()){ if(lui->e3->text().isEmpty()){ bool c2; int val2=ui->e3->text().toInt(&c2); if(!c2){ QMessageBox::critical(NULL,"错误","输入的部长ID不是数字，创建失败!"); } QSqlQuery query; query.prepare("INSERT INTO department (departmentID,departmentname,employeeID) VALUES (:value1,:value2,:value3)"); query.bindValue(":value1",val); query.bindValue(":value2", ui->e2->text()); query.bindValue(":value3",val2); if (!query.exec()) { QMessageBox::critical(NULL,"错误","数据库错误，插入失败，请检查输入!"); return; } } else{ QSqlQuery query; query.prepare("INSERT INTO department (departmentID,departmentname) VALUES (:value1,:value2)"); query.bindValue(":value1",val); query.bindValue(":value2", ui->e2->text()); if (!query.exec()) { QMessageBox::critical(NULL,"错误","数据库错误，插入失败，请检查输入!"); return; } } } else{ if(lui->e3->text().isEmpty()){ bool c2; int val2=ui->e3->text().toInt(&c2); if(!c2){ QMessageBox::critical(NULL,"错误","输入的部长ID不是数字，创建失败!"); } QSqlQuery query; query.prepare("INSERT INTO department (departmentID,employeeID) VALUES (:value1,:value2)"); query.bindValue(":value1",val); query.bindValue(":value2",val2); if (!query.exec()) { QMessageBox::critical(NULL,"错误","数据库错误，插入失败，请检查输入!"); return; } } } } </pre>	

```
        }
        }
        else{
            QSqlQuery query;
            query.prepare("INSERT INTO department (departmentID) VALUES (:value1)");
            query.bindValue(":value1",val);
            if (!query.exec()) {
                QMessageBox::critical(NULL,"错误","数据库错误, 插入失败, 请检查输入!");
                return;
            }
        }
    }
    QMessageBox::information(NULL,"操作完成","操作完成!");
}
```

触发器源码
(3分)

```
mysql> delimiter //
mysql> create trigger checkdpid before insert on department for each row
-> begin
-> if new.departmentID>=10000 or new.departmentID<=1000 then
-> signal sqlstate '45000' set message_text='departmentID must be between 1001 and 9999';
-> end if;
-> end //

mysql> delimiter //
mysql> create trigger updated after insert on department for each row
-> begin
-> update employee set departmentID=new.departmentID where employeeID=new.employeeID;
-> end //
```

程序的初始状态如下。

```
mysql> select * from employee;
```

employeeID	employname	enrollmentdate	salary	job	sex	departmentID	pwd
10001	bob	0	NULL	0	1	1001	10001
10003	cathy	0	NULL	Sleeping	0	1001	10003
10004	cat	NULL	NULL	NULL	1	1005	NULL
10008	Early8	NULL	NULL	NULL	NULL	1005	NULL
10010	Apr26	NULL	NULL	NULL	1	9998	NULL

5 rows in set

```
mysql> select * from department;
```

departmentID	departname	employeeID
0	NULL	NULL
1001	武装部	10001
1005	舞蹈部	NULL
9998	数据库部	10010
9999	NULL	NULL

5 rows in set

添加一个新部门 departmentID: 9990。

程序演示
(4分)



添加一个指定部长名和部长 ID 的部门 departmentID: 9988。

Iwork Admin

2024-05-22 09:34:16

Welcome: admin

部门添加

部门ID

9988

部门名

相声部

部长ID

10008

提交

注:

1.除部门ID外,其他均非必填项。

2.部门ID不可与已有部门ID重复,且必须在001-9999范围内。

3.部长必须存在,且设置部长后,部长的部门自动迁移至新部门。

操作完成

操作完成!

OK

部门修改/移除

部门ID

部门名

null

部长ID

null

新部门ID

修改

移除

注:

1.部门移除操作必须填写部门ID,若填写新部门ID则其中员工迁移至新部门,否则迁移至默认(ID: 0)部门。

2.部门修改操作是通过给定的部门ID修改部门名和部长,部门ID不可缺省,部门名和部长若仅需修改一处,则另一处无需填写,若需改为空值则必须重置该值。重置时请先清空输入。

返回

程序正常执行,此时表中内容如下。

```
mysql> select * from employee;
```

employeeID	employname	enrollmentdate	salary	job	sex	departmentID	pwd
10001	bob	0	NULL	0	1	1001	10001
10003	cathy	0	NULL	Sleeping	0	1001	10003
10004	cat	NULL	NULL	NULL	1	1005	NULL
10008	Early8	NULL	NULL	NULL	NULL	9988	NULL
10010	Apr26	NULL	NULL	NULL	1	9998	NULL

5 rows in set

```
mysql> select * from department;
```

departmentID	departname	employeeID
0	NULL	NULL
1001	武装部	10001
1005	舞蹈部	NULL
9988	相声部	10008
9990	NULL	NULL
9998	数据库部	10010
9999	NULL	NULL

7 rows in set

可以看到,两个新的部门已经完成了创建, employee 表中 employeeID: 10008 的员工部门也转移到了新部门。

在上一部分的基础上,添加一个 departmentID=10005 的部门, 和一个部长 ID 不存在 (10088) 的部门, 结果如下。

程序演示
(4 分)

Iwork Admin

2024-05-22 09:42:08

Welcome: admin

部门添加

部门ID

10005

部门名

游戏部

错误

部长ID

提交

注:

1.除部门ID外,其他均非必填项。

2.部门ID不可与已有部门ID重复,且必须在001-9999范围内。

3.部长必须存在,且设置部长后,部长的部门自动迁移至新部门。

数据库错误, 插入失败, 请检查输入!

OK

部门修改/移除

部门ID

部门名

null

部长ID

null

新部门ID

修改

移除

注:

1.部门移除操作必须填写部门ID,若填写新部门ID则其中员工迁移至新部门,否则迁移至默认(ID: 0)部门。

2.部门修改操作是通过给定的部门ID修改部门名和部长,部门ID不可缺省,部门名和部长若仅需修改一处,则另一处无需填写,若需改为空值则必须重置该值。重置时请先清空输入。

返回

可以看到都成功返回了错误，数据库表也没能得到更新，以下是当前数据库表的结果。

```
mysql> select * from employee;
+-----+-----+-----+-----+-----+-----+-----+
| employeeID | employname | enrollmentdate | salary | job | sex | departmentID | pwd |
+-----+-----+-----+-----+-----+-----+-----+
| 10001 | bob | 0 | NULL | 0 | 1 | 1001 | 10001 |
| 10003 | cathy | 0 | NULL | Sleeping | 0 | 1001 | 10003 |
| 10004 | cat | NULL | NULL | NULL | 1 | 1005 | NULL |
| 10008 | Early8 | NULL | NULL | NULL | NULL | 9988 | NULL |
| 10010 | Apr26 | NULL | NULL | NULL | 1 | 9998 | NULL |
+-----+-----+-----+-----+-----+-----+-----+
5 rows in set

mysql> select * from department;
+-----+-----+-----+
| departmentID | departname | employeeID |
+-----+-----+-----+
| 0 | NULL | NULL |
| 1001 | 武装部 | 10001 |
| 1005 | 舞蹈部 | NULL |
| 9988 | 相声部 | 10008 |
| 9990 | NULL | NULL |
| 9998 | 数据库部 | 10010 |
| 9999 | NULL | NULL |
+-----+-----+-----+
7 rows in set
```

备注	无
----	---

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)	<p>该操作是添加新员工/员工信息修改的一个子模块, 对其 techlevel 和 managelevel 进行设定。然而, 由于这两个参数分别存在 technician 和 management 两张 employee 的子表中。这个操作涉及到的 sql 语句流程较为繁琐, 对其设置存储过程以包装减少 sql 语句传递数量以及增强模块复用性是很有必要的。</p>

存储过程功能描述 (1分)	<p>该存储过程一共传入 7 个参数，依次为 employeeID，是否清除职级的两个复选框结果，是否获得合理修改数据（此处仅仅指的是其是否可以被转换成 int 型变量被传入存储过程）的两个 bool 型变量以及修改数据（如有）。</p> <p>该存储过程包括了表的添加，更新，删除。</p> <p>对于清除操作（即清除复选框为 true），因为其作为子表，应当直接删除表。</p> <p>对于修改操作，若其之前没有该数值，即子表中没有该条目，需要执行添加操作，若已经存在，则应该执行更新操作。</p>	
涉及的关系表 (2分)	employee（作为外键约束），technicist，management	
表连接涉及字段 (1分)	employee.employeeID=technicist.employeeID employee.employeeID=management.employeeID	
更改字段 (2分)	字段	规则
	technicist.techlevel	范围 1-99 数字。
	management.managelevel	范围 1-99 数字。
更新代码 (3分)	<pre> bool c1; int val=ui->u1->text().toInt(&c1); if(!c1){ QMessageBox::critical(NULL,"错误","输入的用户ID不是数字，修改失败！"); return; } bool c8=false; int val8=0; if(!ui->u8->text().isEmpty()){ val8=ui->u8->text().toInt(&c8); if(!c8){ QMessageBox::critical(NULL,"错误","输入的管理职级不是数字！"); } } bool c7=false; int val7=0; if(!ui->u7->text().isEmpty()){ val7=ui->u7->text().toInt(&c7); if(!c7){ QMessageBox::critical(NULL,"错误","输入的技术职级不是数字！"); } } QString query; query.prepare("CALL updatetm(:param1,:param2,:param3,:param4,:param5,:param6,:param7)"); query.bindValue(":param1", val); query.bindValue(":param2", ui->cb7->isChecked()); query.bindValue(":param3", ui->cb8->isChecked()); query.bindValue(":param4", c7); query.bindValue(":param5", c8); query.bindValue(":param6", val7); query.bindValue(":param7", val8); if(!query.exec()) QMessageBox::critical(NULL,"错误","修改技术职级/管理职级失败！"); </pre>	
创建存储过程源码 (3分)	<pre> mysql> delimiter // mysql> create procedure updatetm (IN ceID int,IN snt TINYINT(1),IN snm TINYINT(1) -> ,IN xgt TINYINT(1),IN xgm TINYINT(1),IN newt SMALLINT,IN newm SMALLINT) -> begin if snt=1 then delete from technicist where employeeID=ceID; -> elseif xgt=1 then if newt<=0 or newt>=100 then signal sqlstate '45000' set message_text='techlevel must be between 1 and 99'; -> elseif ceID in (select employeeID from technicist) then -> update technicist set techlevel=newt where employeeID=ceID; -> else INSERT INTO technicist(employeeID,techlevel) VALUES (ceID,newt); -> end if;end if; -> if snm=1 then delete from management where employeeID=ceID; -> elseif xgm=1 then if newm<=0 or newm>=100 then signal sqlstate '45000' set message_text='managelevel must be between 1 and 99'; -> elseif ceID in (select employeeID from management) then -> update management set managelevel=newm where employeeID=ceID; -> else INSERT INTO management(employeeID,managelevel) VALUES (ceID,newm); -> end if; end if;end // Query OK, 0 rows affected mysql> delimiter ; </pre>	

存储过程
执行源码
(1 分)

```
QSqlQuery query;
query.prepare("CALL updatetm(:param1,:param2,:param3,:param4,:param5,:param6,:param7)");
query.bindValue(":param1", val);
query.bindValue(":param2", ui->cb7->isChecked());
query.bindValue(":param3", ui->cb8->isChecked());
query.bindValue(":param4", c7);
query.bindValue(":param5", c8);
query.bindValue(":param6", val7);
query.bindValue(":param7", val8);
if(!query.exec())
    QMessageBox::critical(NULL,"错误","修改技术职级/管理职级失败!");
```

程序演示
(2 分)

在执行更改之前，三个表的状态如下所示。

```
mysql> select * from management;
```

employeeID	managelevel
10001	55
10004	2
10086	6

3 rows in set

```
mysql> select * from technician;
```

techlevel	employeeID
55	10001
2	10004
62	10037
4	10086

4 rows in set

```
mysql> select * from employee;
```

employeeID	employname	enrollmentdate	salary	job	sex	departmentID	pwd
10001	bob	0	NULL	0	1	1001	10001
10003	cathy	0	NULL	Sleeping	0	1001	10003
10004	cat	NULL	NULL	NULL	1	1005	NULL
10008	Early8	NULL	NULL	NULL	NULL	9988	NULL
10010	Apr26	NULL	NULL	NULL	1	9998	NULL
10037	ryan	NULL	NULL	NULL	NULL	NULL	NULL
10086	Helln	2005-03-08	38888	OPERATOR	0	NULL	LAST

7 rows in set

以下主要对更新分支进行演示，首先将 10037 工号的两个职级更改为 10 和 11（本质上是一次添加和一次更新），并将 10086 工号的两个职级均更改为 86（两次更新）。

Iwork Admin

2024-05-22 15:22:54

Welcome: admin

工号

10037

姓名

性别

入职时间

职务

部门ID

☐ null

技术职级

10

☐ null

管理职级

11

☐ null

工资

☐ null

密码

☐ null

操作完成

操作完成!

OK

注:

1.工号必须输入且为数字，其他可以缺省。以工号来选择修改和删除，删除只参照工号。

2.若输入性别，男输入1，女输入0。

3.若设定部门ID，则部门ID必须存在。

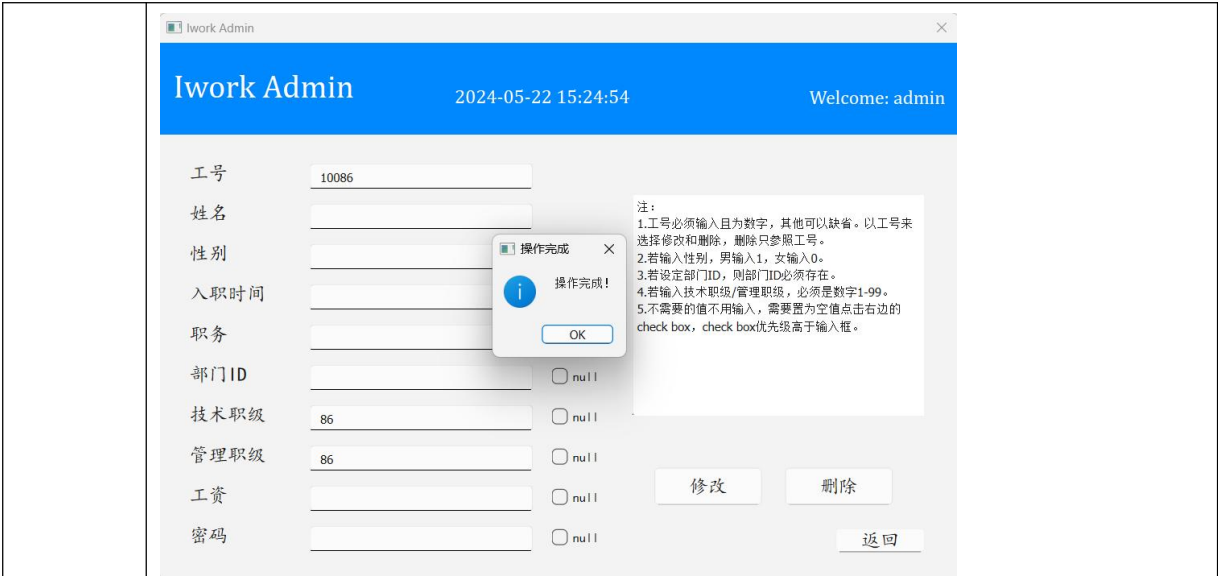
4.若输入技术职级/管理职级，必须是数字1-99。

5.不需要的值不用输入，需要置为空值点击右边的checkbox，checkbox优先级高于输入框。

修改

删除

返回



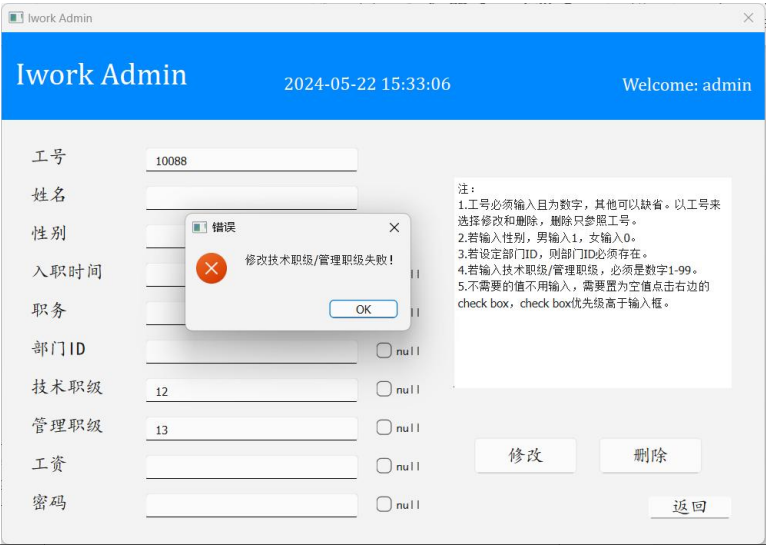
执行后的结果如下。可以看到成功完成了更新。

```
mysql> select * from management;
+-----+-----+
| employeeID | managelevel |
+-----+-----+
| 10001      | 55          |
| 10004      | 2           |
| 10037      | 11          |
| 10086      | 86          |
+-----+-----+
4 rows in set

mysql> select * from technician;
+-----+-----+
| techlevel | employeeID |
+-----+-----+
| 55        | 10001      |
| 2         | 10004      |
| 10        | 10037      |
| 86        | 10086      |
+-----+-----+
4 rows in set
```

程序演示
(2分)

在此基础上，我们再尝试将 10088 工号的员工的职级修改为 12 和 13（违反外键约束），和将 10004 工号的员工的职级修改为-2 和 98（违反存储过程而自动返回错误）。



描述(1分)	加到表单中显示出来。																																																																																																																																																																																														
视图功能描述(1分)	在访问 employee 的两个子表的时候经常需要进行表连接或者嵌套查询操作，为降低该操作的复杂度，我们建立一个视图，将 employee 的两个子表与其进行左连接，可以较好的完成该任务。																																																																																																																																																																																														
涉及的关系表(2分)	employee, management, technician																																																																																																																																																																																														
表连接字段 (1分)	employee.employeeID=management.employeeID employee.employeeID=technician.employeeID																																																																																																																																																																																														
创建视图代码(3分)	<pre>mysql> create view fullemployee as -> select employee.*,management.managelevel,technician.techlevel from employee -> left join management on employee.employeeID=management.employeeID -> left join technician on employee.employeeID=technician.employeeID; Query OK, 0 rows affected</pre>																																																																																																																																																																																														
查询代码 (3分)	<pre>QString ct = ui->Box->currentText(); if(ct.isEmpty()){ QMessageBox::information(NULL,"注意","您没有选择合适的部门"); return; } while (ui->widget->rowCount()) ui->widget->removeRow(0); QStringList parts = ct.split('-'); QSqlQuery query; int search=parts[0].toInt(); if(search) query.prepare("SELECT * FROM fullemployee WHERE departmentID = :departmentID ORDER BY employeeID"); else query.prepare("SELECT * FROM fullemployee WHERE departmentID = :departmentID OR departmentID IS NULL ORDER BY employeeID"); query.bindValue(":departmentID", search); query.exec(); while(query.next()){ int row = ui->widget->rowCount(); ui->widget->insertRow(row); ui->widget->setItem(row, 0, new QTableWidgetItem(query.value("employeeID").toString())); ui->widget->setItem(row, 1, new QTableWidgetItem(query.value("employname").toString())); if(!query.value("sex").isNull()) ui->widget->setItem(row, 2, new QTableWidgetItem((query.value("sex").toBool()==true?"男":"女"))); ui->widget->setItem(row, 3, new QTableWidgetItem(query.value("enrollmentdate").toString())); ui->widget->setItem(row, 4, new QTableWidgetItem(query.value("job").toString())); ui->widget->setItem(row, 5, new QTableWidgetItem(query.value("techlevel").toString())); ui->widget->setItem(row, 6, new QTableWidgetItem(query.value("managelevel").toString())); ui->widget->setItem(row, 7, new QTableWidgetItem(query.value("salary").toString())); ui->widget->setItem(row, 8, new QTableWidgetItem(query.value("pwd").toString())); }</pre>																																																																																																																																																																																														
程序演示 (4分)	<p>下图是视图的初始情况，为了测试已经预先插入了一些数据。</p> <pre>mysql> select * from fullemployee -> ;</pre> <table><thead><tr><th>employeeID</th><th>employname</th><th>enrollmentdate</th><th>salary</th><th>job</th><th>sex</th><th>departmentID</th><th>pwd</th><th>managelevel</th><th>techlevel</th></tr></thead><tbody><tr><td>10001</td><td>bob</td><td>0</td><td>NULL</td><td>0</td><td>1</td><td>1001</td><td>10001</td><td>55</td><td>55</td></tr><tr><td>10003</td><td>cathy</td><td>0</td><td>NULL</td><td>Sleeping</td><td>0</td><td>1001</td><td>10003</td><td>NULL</td><td>NULL</td></tr><tr><td>10004</td><td>cat</td><td>NULL</td><td>NULL</td><td>NULL</td><td>1</td><td>1005</td><td>NULL</td><td>2</td><td>2</td></tr><tr><td>10008</td><td>Early8</td><td>NULL</td><td>NULL</td><td>NULL</td><td>NULL</td><td>9988</td><td>NULL</td><td>NULL</td><td>NULL</td></tr><tr><td>10010</td><td>Apr26</td><td>NULL</td><td>NULL</td><td>NULL</td><td>1</td><td>9998</td><td>NULL</td><td>NULL</td><td>NULL</td></tr><tr><td>10037</td><td>ryan</td><td>NULL</td><td>NULL</td><td>NULL</td><td>NULL</td><td>NULL</td><td>NULL</td><td>11</td><td>10</td></tr><tr><td>10086</td><td>Helin</td><td>2005-03-08</td><td>38888</td><td>OPERATOR</td><td>0</td><td>NULL</td><td>LAST</td><td>86</td><td>86</td></tr><tr><td>11000</td><td>test</td><td>NULL</td><td>NULL</td><td>NULL</td><td>NULL</td><td>1001</td><td>NULL</td><td>NULL</td><td>NULL</td></tr><tr><td>11001</td><td>test</td><td>NULL</td><td>NULL</td><td>NULL</td><td>NULL</td><td>1001</td><td>NULL</td><td>NULL</td><td>NULL</td></tr><tr><td>11002</td><td>test</td><td>NULL</td><td>NULL</td><td>NULL</td><td>NULL</td><td>1001</td><td>NULL</td><td>NULL</td><td>NULL</td></tr><tr><td>11003</td><td>test</td><td>NULL</td><td>NULL</td><td>NULL</td><td>NULL</td><td>1001</td><td>NULL</td><td>NULL</td><td>NULL</td></tr><tr><td>11004</td><td>test</td><td>NULL</td><td>NULL</td><td>NULL</td><td>NULL</td><td>1001</td><td>NULL</td><td>NULL</td><td>NULL</td></tr><tr><td>11005</td><td>test</td><td>NULL</td><td>NULL</td><td>NULL</td><td>NULL</td><td>1001</td><td>NULL</td><td>NULL</td><td>NULL</td></tr><tr><td>11006</td><td>test</td><td>NULL</td><td>NULL</td><td>NULL</td><td>NULL</td><td>1001</td><td>NULL</td><td>NULL</td><td>NULL</td></tr><tr><td>11007</td><td>test</td><td>NULL</td><td>NULL</td><td>NULL</td><td>NULL</td><td>1001</td><td>NULL</td><td>NULL</td><td>NULL</td></tr><tr><td>11008</td><td>test</td><td>NULL</td><td>NULL</td><td>NULL</td><td>NULL</td><td>1001</td><td>NULL</td><td>NULL</td><td>NULL</td></tr><tr><td>11009</td><td>test</td><td>NULL</td><td>NULL</td><td>NULL</td><td>NULL</td><td>1001</td><td>NULL</td><td>NULL</td><td>NULL</td></tr><tr><td>11010</td><td>test</td><td>NULL</td><td>NULL</td><td>NULL</td><td>NULL</td><td>1001</td><td>NULL</td><td>NULL</td><td>NULL</td></tr></tbody></table> <pre>18 rows in set</pre> <p>以下给出三个表的查询结果与之对照。</p>	employeeID	employname	enrollmentdate	salary	job	sex	departmentID	pwd	managelevel	techlevel	10001	bob	0	NULL	0	1	1001	10001	55	55	10003	cathy	0	NULL	Sleeping	0	1001	10003	NULL	NULL	10004	cat	NULL	NULL	NULL	1	1005	NULL	2	2	10008	Early8	NULL	NULL	NULL	NULL	9988	NULL	NULL	NULL	10010	Apr26	NULL	NULL	NULL	1	9998	NULL	NULL	NULL	10037	ryan	NULL	NULL	NULL	NULL	NULL	NULL	11	10	10086	Helin	2005-03-08	38888	OPERATOR	0	NULL	LAST	86	86	11000	test	NULL	NULL	NULL	NULL	1001	NULL	NULL	NULL	11001	test	NULL	NULL	NULL	NULL	1001	NULL	NULL	NULL	11002	test	NULL	NULL	NULL	NULL	1001	NULL	NULL	NULL	11003	test	NULL	NULL	NULL	NULL	1001	NULL	NULL	NULL	11004	test	NULL	NULL	NULL	NULL	1001	NULL	NULL	NULL	11005	test	NULL	NULL	NULL	NULL	1001	NULL	NULL	NULL	11006	test	NULL	NULL	NULL	NULL	1001	NULL	NULL	NULL	11007	test	NULL	NULL	NULL	NULL	1001	NULL	NULL	NULL	11008	test	NULL	NULL	NULL	NULL	1001	NULL	NULL	NULL	11009	test	NULL	NULL	NULL	NULL	1001	NULL	NULL	NULL	11010	test	NULL	NULL	NULL	NULL	1001	NULL	NULL	NULL
employeeID	employname	enrollmentdate	salary	job	sex	departmentID	pwd	managelevel	techlevel																																																																																																																																																																																						
10001	bob	0	NULL	0	1	1001	10001	55	55																																																																																																																																																																																						
10003	cathy	0	NULL	Sleeping	0	1001	10003	NULL	NULL																																																																																																																																																																																						
10004	cat	NULL	NULL	NULL	1	1005	NULL	2	2																																																																																																																																																																																						
10008	Early8	NULL	NULL	NULL	NULL	9988	NULL	NULL	NULL																																																																																																																																																																																						
10010	Apr26	NULL	NULL	NULL	1	9998	NULL	NULL	NULL																																																																																																																																																																																						
10037	ryan	NULL	NULL	NULL	NULL	NULL	NULL	11	10																																																																																																																																																																																						
10086	Helin	2005-03-08	38888	OPERATOR	0	NULL	LAST	86	86																																																																																																																																																																																						
11000	test	NULL	NULL	NULL	NULL	1001	NULL	NULL	NULL																																																																																																																																																																																						
11001	test	NULL	NULL	NULL	NULL	1001	NULL	NULL	NULL																																																																																																																																																																																						
11002	test	NULL	NULL	NULL	NULL	1001	NULL	NULL	NULL																																																																																																																																																																																						
11003	test	NULL	NULL	NULL	NULL	1001	NULL	NULL	NULL																																																																																																																																																																																						
11004	test	NULL	NULL	NULL	NULL	1001	NULL	NULL	NULL																																																																																																																																																																																						
11005	test	NULL	NULL	NULL	NULL	1001	NULL	NULL	NULL																																																																																																																																																																																						
11006	test	NULL	NULL	NULL	NULL	1001	NULL	NULL	NULL																																																																																																																																																																																						
11007	test	NULL	NULL	NULL	NULL	1001	NULL	NULL	NULL																																																																																																																																																																																						
11008	test	NULL	NULL	NULL	NULL	1001	NULL	NULL	NULL																																																																																																																																																																																						
11009	test	NULL	NULL	NULL	NULL	1001	NULL	NULL	NULL																																																																																																																																																																																						
11010	test	NULL	NULL	NULL	NULL	1001	NULL	NULL	NULL																																																																																																																																																																																						

```
mysql> select * from employee
-> ;
```

employeeID	employname	enrollmentdate	salary	job	sex	departmentID	pwd
10001	bob	0	NULL	0	1	1001	10001
10003	cathy	0	NULL	Sleeping	0	1001	10003
10004	cat	NULL	NULL	NULL	1	1005	NULL
10008	Early8	NULL	NULL	NULL	NULL	9988	NULL
10010	Apr26	NULL	NULL	NULL	1	9998	NULL
10037	ryan	NULL	NULL	NULL	NULL	NULL	NULL
10086	Helln	2005-03-08	38888	OPERATOR	0	NULL	LAST
11000	test	NULL	NULL	NULL	NULL	1001	NULL
11001	test	NULL	NULL	NULL	NULL	1001	NULL
11002	test	NULL	NULL	NULL	NULL	1001	NULL
11003	test	NULL	NULL	NULL	NULL	1001	NULL
11004	test	NULL	NULL	NULL	NULL	1001	NULL
11005	test	NULL	NULL	NULL	NULL	1001	NULL
11006	test	NULL	NULL	NULL	NULL	1001	NULL
11007	test	NULL	NULL	NULL	NULL	1001	NULL
11008	test	NULL	NULL	NULL	NULL	1001	NULL
11009	test	NULL	NULL	NULL	NULL	1001	NULL
11010	test	NULL	NULL	NULL	NULL	1001	NULL

18 rows in set

```
mysql> select * from technician;
```

techlevel	employeeID
55	10001
2	10004
10	10037
86	10086

4 rows in set

```
mysql> select * from management;
```

employeeID	managelevel
10001	55
10004	2
10037	11
10086	86

4 rows in set

可以看到，视图的创建是符合我们目标的，接下来进行前端演示。
以对无部门 ID（或部门 ID 为 0）员工和部门 ID 为 1001 的员工进行验证。

Iwork Admin
2024-05-23 00:31:30
Welcome: admin

部门ID
0-
查询
返回上层

工号	姓名	性别	入职时间	职务	技术	管理	工资	密码
1 10037	ryan				10	11		
2 10086	Helln	女	2005-03-08	OPERATOR	86	86	38888	LAST

	<div><div>Iwork Admin</div><div><div>Iwork Admin</div><div>2024-05-23 00:29:39</div><div>Welcome: admin</div></div><div><div><div>部门ID</div><div>1001-武装部</div><div>0-</div><div>1001-武装部</div><div>1005-舞蹈部</div><div>9988-相声部</div><div>9990-</div><div>9998-数据库部</div><div>9999-</div></div><div><div>工号</div><div>10001</div><div>10003</div><div>11000</div><div>11001</div><div>11002</div><div>11003</div><div>11004</div><div>11005</div><div>11006</div><div>11007</div></div><div><div>bob</div><div>cat</div><div>tes</div><div>test</div><div>test</div><div>test</div><div>test</div><div>test</div><div>test</div><div>test</div></div><div><div>业务</div><div>技术</div><div>管理</div><div>工资</div><div>密码</div></div><div><div>55</div><div>55</div><div>0</div><div>0</div><div>0</div><div>0</div><div>0</div><div>0</div><div>0</div><div>0</div></div><div><div>10001</div><div>10003</div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div></div></div></div> <div>可以看到，各组件均显示了正确的结果。</div>
备注	无