

数据库系统课程实验报告

|  |  |
| --- | --- |
| 实验名称： | 实验七 数据库完整性 |
| 实验日期： | 2023-5-19 |
| 实验地点： | 文宣楼B311 |
| 提交日期： | 2023-5-19 |
|  | |
| 学号： | 33920212204567 |
| 姓名： | 任宇 |
| 专业年级： | 软工2021级 |
| 学年学期： | 2022-2023学年第二学期 |

1. 实验环境

* 华为 ECS+openGauss 数据库服务器平台

前提：openGauss 数据库服务器正常运行

1. 实验目的

* 理解并掌握关系数据库完整性的运行机制

■ 完整性约束定义>完整性约束检查>违约处理

* 理解并掌握关系数据库完整性主要约束类型及其含义和作用

■ PRIMARY KEY，FOREIGN KEY，NOT NULL，UNIQUE，CHECK

* 理解并掌握关系数据库完整性定义、修改、删除和重命名的方法

■ CREATE TABLE，ALTER TABLE

* 熟练掌握 openGauss 下通过系统表 pg\_constraint 查看完整性信息的方法
* 熟练掌握 openGauss 下通过查看表结构来查看主外码信息的方法
* 熟练掌握 openGauss 下通过查看完整性约束定义的方法

1. 实验内容和步骤
2. 创建两张表：雇员表 Emp 和工作表 Work：

**CREATE TABLE Emp (**

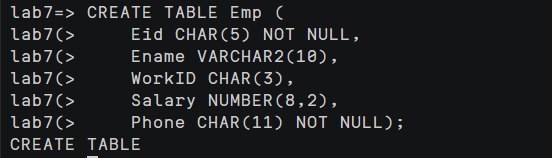
**Eid CHAR(5) NOT NULL,**

**Ename VARCHAR2(10),**

**WorkID CHAR(3),**

**Salary NUMBER(8,2),**

**Phone CHAR(11) NOT NULL);**

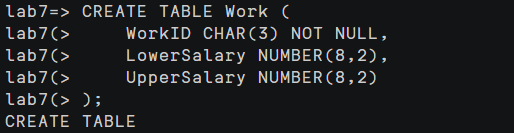


**CREATE TABLE Work (**

**WorkID CHAR(3) NOT NULL,**

**LowerSalary NUMBER(8,2),**

**UpperSalary NUMBER(8,2));**



1. 分别为两张表插入如下数据，查看插入操作是否成功。

雇员表数据：{( '10001 ', 'Smith ', '001 ', 2000, '13800010001 '),( '10001 ', 'Jonny ', '001 ', 3000, '13600010002 ')，

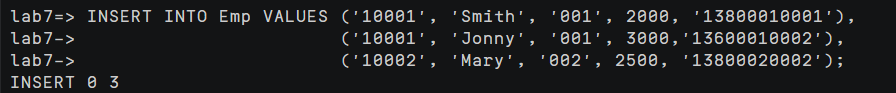
('10002 ', 'Mary ', '002 ', 2500, '13800020002 ') }

工作表数据：{ ('001 ', 1000, 5000)，('002 ', 2000, 8000) }：

**INSERT INTO Emp VALUES ('10001', 'Smith', '001', 2000, '13800010001'),**

**('10001', 'Jonny', '001', 3000,'13600010002'),**

**('10002', 'Mary', '002', 2500, '13800020002');**



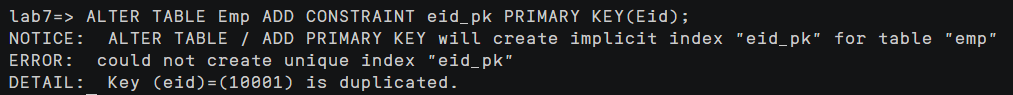
**INSERT INTO Work VALUES ('001', 1000, 5000),('002', 2000, 8000);**



两表都插入成功

1. 修改雇员表的结构，设置 Eid 为主码，主码名称为 eid\_pk，查看该操作是否成功。若不成功，请说明原因并思考如何处理才能成功添加约束。要求：所有约束都要显式给出约束名，不可由系统默认，因为删除约束时需要用到约束名:

**ALTER TABLE Emp ADD CONSTRAINT eid\_pk PRIMARY KEY(Eid);**



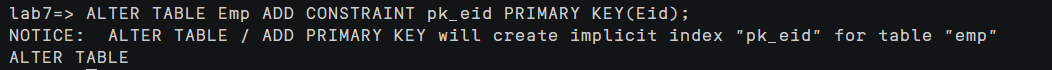
操作失败，因为主码值应该唯一，但出现了两个10001，把其中一个修改为10003

**UPDATE Emp SET Eid = '10003' WHERE Ename = 'Jonny';**



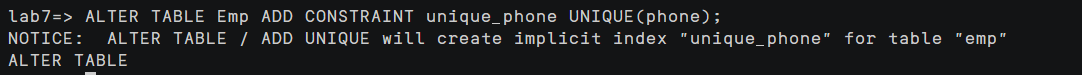
1. 将 eid 为主码的约束名 eid\_pk 改为 pk\_eid:

**ALTER TABLE Emp ADD CONSTRAINT pk\_eid PRIMARY KEY(Eid);**



1. 设置雇员表中的 phone 字段取唯一值，查看该操作是否成功？若不成功说明原因:

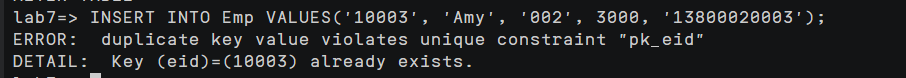
**ALTER TABLE Emp ADD CONSTRAINT unique\_phone UNIQUE(phone);**



操作成功

1. 给雇员表添加一条新记录('10003', 'Amy', '002', 3000, '13800020003')，查看执行结果:

**INSERT INTO Emp VALUES('10003', 'Amy', '002', 3000, '13800020003');**



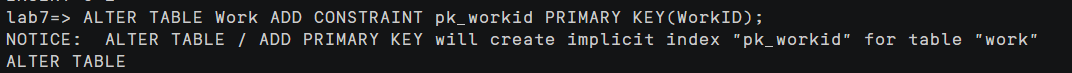
操作失败，因为当前表中已经存在10003的主码键，把这条纪录中的10003改为10004后插入，操作成功

**INSERT INTO Emp VALUES('10004', 'Amy', '002', 3000, '13800020003');**



1. 设置工作表的 WorkID 为主码:

**ALTER TABLE Work ADD CONSTRAINT pk\_workid PRIMARY KEY(WorkID);**



1. 修改雇员表，设置雇员表的 WorkID 字段为外码，它引用工作表中的 WorkID 字段，查看操作是否成功？若不成功说明原因:

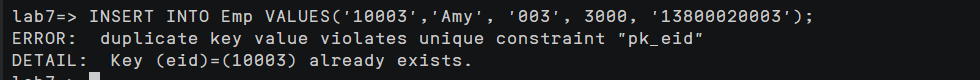
**ALTER TABLE Emp ADD CONSTRAINT fk\_workid FOREIGN KEY(WorkID) REFERENCES Work(WorkID);**



操作成功

1. 给雇员表添加一条新记录('10003', 'Amy', '003', 3000, '13800020003')，查看操作是否成功？若不成功说明原因:

**INSERT INTO Emp VALUES('10003','Amy', '003', 3000, '13800020003');**



操作失败，原因如下：（1）主码值10003已经存在表中，主码值应该唯一（2）phone值应该唯一，而新增数据的电话已经存在（3）根据参照性完整性，外码或为空，或者为被参照表中的值，而003不在被参照表的值中。

1. 在雇员表中，设置雇员工资必须大于或等于 1000。查看操作是否成功？若不成功说明原因:

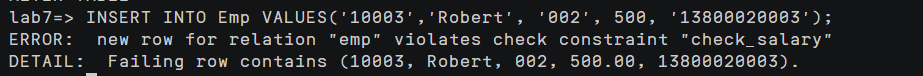
**ALTER TABLE Emp ADD CONSTRAINT check\_salary CHECK(Salary >= 1000);**



操作成功

1. 给雇员表添加一条新记录('10003', 'Robert', '002',500, '13800020003')，查看执行操作是否成功？若不成功说明原因:

**INSERT INTO Emp VALUES('10003','Robert', '002', 500, '13800020003');**



操作失败，原因如下：（1）根据约束check\_salary,salary的值应该大于或等于1000，新增值为500，不满足约束条件（2）主码值10003已经存在，不满足主码的约束条件。

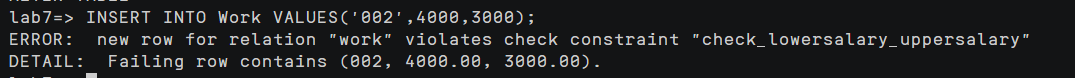
1. 在工作表中，设置其最低工资不超过最高工资:

**ALTER TABLE Work ADD CONSTRAINT check\_LowerSalary\_UpperSalary CHECK(LowerSalary <= UpperSalary);**



1. 给工作表添加一条新记录('002',4000,3000)，查看操作是否成功？若不成功说明原因：

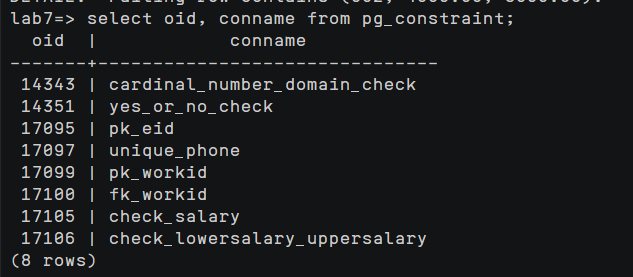
**INSERT INTO Work VALUES('002',4000,3000);**



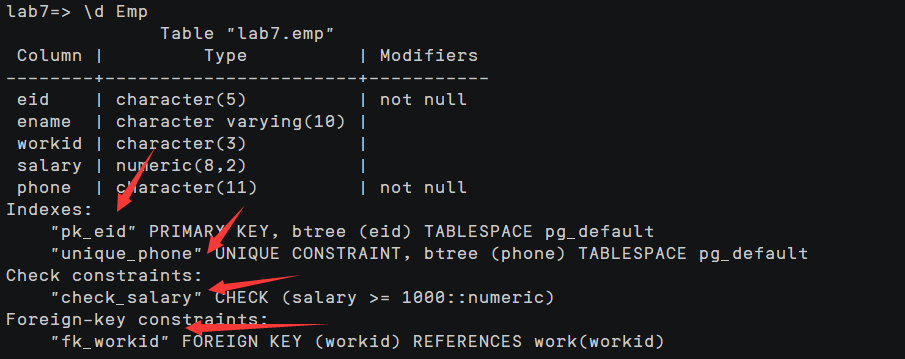
操作失败，原因：插入的值中，最低工资4000大于最高工资3000，不满足最低工资不超过最高工资的约束条件，操作失败。

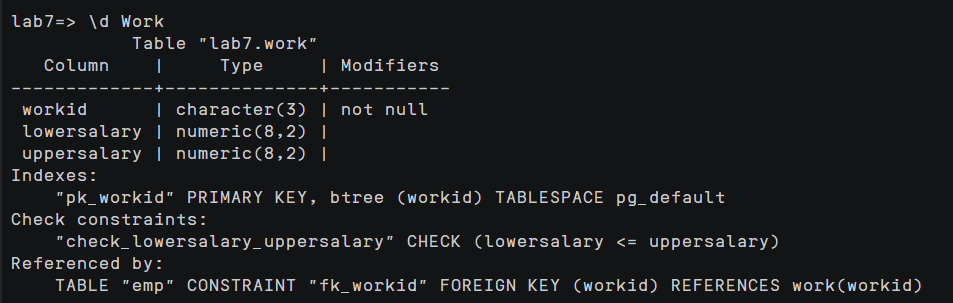
1. 通过查看 openGauss 的系统表 pg\_constraint 了解表上的约束：

**select oid, conname from pg\_constraint;**



1. 通过 gsql 命令\d+ table\_name 查看改表上的约束定义：





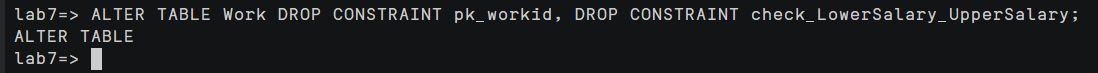
1. 删除雇员表的所有约束，包括主码约束、外码约束和其他约束：

**ALTER TABLE Emp DROP CONSTRAINT pk\_eid, DROP CONSTRAINT unique\_phone,DROP CONSTRAINT check\_salary, DROP CONSTRAINT fk\_workid;**



1. 删除工作表所有约束，包括主码约束：

**ALTER TABLE Work DROP CONSTRAINT pk\_workid, DROP CONSTRAINT check\_LowerSalary\_UpperSalary;**



1. 实验总结

4.1 完成的工作

1. 创建表，并对创建的表进行了添加约束、修改约束、删除约束等操作。

4.2 对实验的认识

通过本次实验，掌握了PRIMARY KEY、FOREIGH KEY、NOT NULL、UNIQUE等约束条件的添加、修改和删除，在操作过程中也学习到应该为设置的约束增加名字，而不是使用系统默认的约束名，以便于进行后续的删除操作。

4.3 遇到的困难及解决方法

无。

1. 实验思考

 openGauss 实现完整性规则的机制是什么？在 SQL 语句中实现完整性规则的常见约束有哪些？各自适用什么业务场景？

答：

机制：（1）提供定义完整性约束条件的机制（2）提供完整性检查的方法（3）进行违约处理

常见约束：

（1）NOT NULL，用于保证该属性的值不能为空。

（2）UNIQUE，唯一约束，用于保证该属性的值有唯一性，可以为空。

（3）PRIMARY KEY，主码，用于保证该属性的值有唯一性，并且非空，是能够唯一的标识一组数据的数据元素。

（4）CHECK，检查约束，按照提供的检查条件检查属性是否符合约束。

（5）FOREIGN KEY，外键，一个表中的 FOREIGN KEY 指向另一个表中的 UNIQUE KEY(唯一约束的键)。参照完整性将两个表中的相应元组联系起来了。