



数据库系统课程实验报告

实验名称:	实验七：数据库的完整性
实验日期:	2024.5.13--
实验地点:	文宣楼 B313
提交日期:	2024.5.18

学号:	37220222203791
姓名:	魏一诺
专业年级:	软工 2022 级
学年学期:	2023-2024 学年第二学期

1.实验目的

- a) 理解并掌握关系数据库完整性的运行机制
- b) 完整性约束定义>完整性约束检查>违约处理
- c) 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
- d) PRIMARY KEY, FOREIGN KEY, NOT NULL, UNIQUE, CHECK
- e) 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
- f) CREATE TABLE, ALTER TABLE
- g) 熟练掌握 openGauss 下通过系统表 pg_constraint 查看完整性信息的方法
- h) 熟练掌握 openGauss 下通过查看表结构来查看主外码信息的方法
- i) 熟练掌握 openGauss 下通过查看完整性约束定义的方法

2.实验内容和步骤

- (1) 创建两张表：雇员表 Emp 和工作表 Work，它们的表结构如下：

Emp 表			
字段	含义	数据类型	是否空
Eid	雇员编号	定长字符型，长度为 5	否
Ename	雇员姓名	变长字符型，长度为 10	/
WorkID	工作编号	定长字符，长度为 3	/
Salary	工资	数值型，总长度为 8，包括两位小数	/
Phone	电话号码	定长字符型，长度为 11	否
Work 表			
字段	含义	数据类型	是否空
WorkID	工作编号	定长字符，长度为 3	否
LowerSalary	最低工资	数值型，总长度为 8，包括两位小数	/
UpperSalary	最高工资	数值型，总长度为 8，包括两位小数	/

步骤如下：

1.

```
cd /opt/software/openGauss/script/  
  
//步骤 1 在数据库主节点服务器上，切换至 omm 操作系统用户  
环境  
  
su - omm  
  
gs_om -t status    //步骤 2 查看服务是否启动  
  
gs_om -t start     //步骤 3 启动数据库服务  
  
gsql -d sales -p 26000 -U wyn -W Bigdata@123 -r //连接到数据库
```

4.创建表

```
Create table Work(WorkID CHAR(3),LowerSalary DECIMAL  
(8,2),UpperSalary DECIMAL (8,2) );
```

```
Create table Emp(Eid CHAR(5) NOT NULL,Ename  
varchar(10),WorkID CHAR(3),Salary DECIMAL(8,2),Phone  
varchar(11) NOT NULL);;
```

```
sales=> Create table Work(WorkID CHAR(3),LowerSalary DECIMAL (8,2),UpperSalary DECIMAL (8,2) );  
CREATE TABLE
```

```
sales=> Create table Emp(Eid CHAR(5) NOT NULL,Ename varchar(10),WorkID CHAR(3),Salary DECIMAL(8,2),Phone varchar(11) NOT NULL);  
CREATE TABLE
```

(2) 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 '),(' 10002 ' , ' Smith ' , ' 002 ' ,2000, ' 13800010002 '),(' 10003 ' , ' Smith ' , ' 003 ' ,2000, ' 13800010003 '),(' 10004 ' , ' Smith ' , ' 004 ' ,2000, ' 13800010004 '),(' 10005 ' , ' Smith ' , ' 005 ' ,2000, ' 13800010005 ')}

10001 ' , ' Jonny ' , ' 001 ' , 3000 , ' 13600010002 ') ,
(' 10002 ' , ' Mary ' , ' 002 ' , 2500 , ' 13800020002 ') }

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

步骤如下:

```
INSERT INTO Emp (Eid, Ename, WorkID, Salary, Phone)
VALUES
    ('10001', 'Smith', '001', 2000, '13800010001'),
    ('10001', 'Jonny', '001', 3000, '13600010002'),
    ('10002', 'Mary', '002', 2500, '13800020002');

INSERT INTO Work (WorkID, LowerSalary, UpperSalary)
VALUES
    ('001', 1000, 5000),
    ('002', 2000, 8000);
```

```
sales=> INSERT INTO Emp (Eid, Ename, WorkID, Salary, Phone)
sales-> VALUES
sales->      ('10001', 'Smith', '001', 2000, '13800010001'),
sales->      ('10001', 'Jonny', '001', 3000, '13600010002'),
sales->      ('10002', 'Mary', '002', 2500, '13800020002');
INSERT 0 3
sales=> INSERT INTO Work (WorkID, LowerSalary, UpperSalary)
sales-> VALUES
sales->      ('001', 1000, 5000),
sales->      ('002', 2000, 8000);
INSERT 0 2
```

(3) 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);
```

```
sales=> ALTER TABLE Emp
sales-> ADD CONSTRAINT eid_pk PRIMARY KEY (Eid);
NOTICE: ALTER TABLE / ADD PRIMARY KEY will create implicit index "eid_pk" for table "emp"
ERROR: could not create unique index "eid_pk"
DETAIL: Key (eid)=(10001) is duplicated.
```

在尝试创建主键约束时，发现了重复的键值（在列 "Eid" 中）。主键约束要求所有值都是唯一的，因此不能有重复值,创建失败

```
sales=> SELECT Eid, COUNT(*)
sales-> FROM Emp
sales-> GROUP BY Eid
sales-> HAVING COUNT(*) > 1;
  eid | count
-----+-----
 10001 |      2
(1 row)
```

查找到了重复的 eid

进行更新：

```
sales=> UPDATE Emp
sales-> SET Eid = '10003'
sales-> WHERE Eid = '10001' AND Ename='Jonny';
UPDATE 1
```

创建成功：

```
sales=> ALTER TABLE Emp
sales-> ADD CONSTRAINT eid_pk PRIMARY KEY (Eid);
NOTICE: ALTER TABLE / ADD PRIMARY KEY will create implicit index "eid_pk" for table "emp"
ALTER TABLE
```

(4) 将 eid 为主码的约束名 eid_pk 改为 pk_eid。

```
sales=> SELECT conname, pg_get_constraintdef(oid)
sales-> FROM pg_constraint
sales-> WHERE conrelid = 'Emp'::regclass;
  conname | pg_get_constraintdef
-----+-----
  eid_pk  | PRIMARY KEY (eid)
(1 row)

sales=> ALTER TABLE Emp DROP CONSTRAINT eid_pk;
ALTER TABLE
sales=> ALTER TABLE Emp ADD CONSTRAINT pk_eid PRIMARY KEY (Eid);
NOTICE: ALTER TABLE / ADD PRIMARY KEY will create implicit index "pk_eid" for table "emp"
ALTER TABLE
```

(5) 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。

```
-- 添加唯一性约束
```

```
ALTER TABLE Emp ADD CONSTRAINT uk_phone UNIQUE  
(Phone);
```

```
sales=> ALTER TABLE Emp ADD CONSTRAINT uk_phone UNIQUE (Phone);  
NOTICE: ALTER TABLE / ADD UNIQUE will create implicit index "uk_phone" for table "emp"  
ALTER TABLE
```

(6) 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 (Eid, Ename, WorkID, Salary, Phone)  
VALUES ('10003', 'Robert', '002', 500, '13800020003');
```

```
sales=> INSERT INTO Emp (Eid, Ename, WorkID, Salary, Phone)  
sales-> VALUES ('10003', 'Robert', '002', 500, '13800020003');  
ERROR: duplicate key value violates unique constraint "pk_eid"  
DETAIL: Key (eid)=(10003) already exists.
```

更新一下：

```
sales=> UPDATE Emp  
sales-> SET Eid='10000'  
sales-> WHERE Eid='10003';  
UPDATE 1
```

插入成功：

```
sales=> INSERT INTO Emp (Eid, Ename, WorkID, Salary, Phone)  
sales-> VALUES ('10003', 'Robert', '002', 500, '13800020003');  
INSERT 0 1
```

(7) 设置工作表的 WorkID 为主码。

```
ALTER TABLE Work  
ADD CONSTRAINT pk_WorkID PRIMARY KEY (WorkID);
```

```
sales=> ALTER TABLE Emp
sales-> ADD CONSTRAINT fk_WorkID FOREIGN KEY (WorkID) REFERENCES Work (WorkID);
ERROR:  there is no unique constraint matching given keys for referenced table "work"
```

(8) 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);
```

```
sales=> ALTER TABLE Emp
sales-> ADD CONSTRAINT fk_WorkID
sales-> FOREIGN KEY (WorkID) REFERENCES Work(WorkID);
ALTER TABLE
```

(9) 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 (Eid, Ename, WorkID, Salary, Phone)

VALUES ('10003', 'Amy', '003', 3000, '13800020003');
```

```
sales=> INSERT INTO Emp (Eid, Ename, WorkID, Salary, Phone)
sales-> VALUES ('10003', 'Amy', '003', 3000, '13800020003');
ERROR:  duplicate key value violates unique constraint "pk_eid"
DETAIL:  Key (eid)=(10003) already exists. 
```

不成功:

因为 Eid 是主键不能重复，已存在 Eid 为 10003 的了

(10) 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 chk_Salary CHECK (Salary >= 1000);
```


不成功:

```
sales-> ADD CONSTRAINT chk_Salary CHECK (Salary >= 1000);
ERROR: check constraint "chk_salary" is violated by some row
```

因为有现有的行不符合这个约束:

```
ERROR: check constraint "chk_salary" is violated
sales=> SELECT *
sales-> FROM Emp
sales-> WHERE Salary < 1000;
   eid |  ename  | workid | salary |   phone
-----+-----+-----+-----+-----
 10003 | Robert |    002 | 500.00 | 13800020003
(1 row)
```

更新一下:

```
sales=> UPDATE Emp
sales-> SET salary='1005'
sales-> WHERE salary='500.00'
sales-> ;
UPDATE 1
```

添加成功:

```
sales=> ALTER TABLE Emp
sales-> ADD CONSTRAINT chk_Salary CHECK (Salary >= 1000);
ALTER TABLE
```

(11) 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 (Eid, Ename, WorkID, Salary, Phone)
VALUES ('10003', 'Robert', '002', 500, '13800020003');
```

```
sales=> INSERT INTO Emp (Eid, Ename, WorkID, Salary, Phone)
sales-> VALUES ('10003', 'Robert', '002', 500, '13800020003');
ERROR: new row for relation "emp" violates check constraint "chk_salary"
DETAIL: Failing row contains (10003, Robert, 002, 500.00, 13800020003).
```

不满足工资要高于 1000 的约束

(12) 在工作表中，设置其最低工资不超过最高工资。

```
ALTER TABLE Work  
  
ADD CONSTRAINT CheckSalaryRange CHECK (LowerSalary <=  
UpperSalary);
```

```
sales-> ADD CONSTRAINT CheckSalaryRange CHECK (LowerSalary <= UpperSalary);  
ALTER TABLE
```

(13) 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 (WorkID, LowerSalary, UpperSalary) VALUES  
( '002', 4000, 3000);
```

```
sales=> INSERT INTO Work (WorkID, LowerSalary, UpperSalary) VALUES ( '002', 4000, 3000);  
ERROR: new row for relation "work" violates check constraint "checksalaryrange"  
DETAIL: Failing row contains (002, 4000.00, 3000.00).
```

试图向表中插入的记录的 LowerSalary 值(4000)大于 UpperSalary 值(3000)，这违反了在表定义中设置的约束，因此插入操作会失败。

(14) 通过查看 openGauss 的系统表 pg_constraint 了解表上的约束。

```
SELECT conname, contype, conkey, consrc  
  
FROM pg_constraint  
  
WHERE conrelid = 'Work'::regclass;
```

```
sales=> SELECT conname, contype, conkey, consrc  
sales-> FROM pg_constraint  
sales-> WHERE conrelid = 'Work'::regclass;  
      conname      | contype | conkey |      consrc  
-----+-----+-----+-----  
pk_workid          | p       | {1}    |  
checksalaryrange  | c       | {2,3}  | (lowersalary <= uppersalary)  
(2 rows)
```

```
SELECT conname, contype, conkey, consrc
FROM pg_constraint
WHERE conrelid = 'Emp'::regclass;
```

```
sales=> SELECT conname, contype, conkey, consrc
sales-> FROM pg_constraint
sales-> WHERE conrelid = 'Emp'::regclass;
   conname   | contype | conkey |          consrc
-----+-----+-----+-----
pk_eid       | p       | {1}    |
uk_phone     | u       | {5}    |
fk_workid    | f       | {3}    |
chk_salary   | c       | {4}    | (salary >= (1000)::numeric)
(4 rows)
```

(15) 通过 `gsql` 命令 `\d+ table_name` 查看该表上的约束定义。

`\d Work`

`\d Emp`

```
sales=> \d Work
          Table "sales.work"
   Column   |      Type      | Modifiers
-----+-----+-----
workid      | character(3)    | not null
lowersalary  | numeric(8,2)    |
uppersalary  | numeric(8,2)    |
Indexes:
    "pk_workid" PRIMARY KEY, btree (workid) TABLESPACE pg_default
Check constraints:
    "checksalaryrange" CHECK (lowersalary <= uppersalary)
Referenced by:
    TABLE "emp" CONSTRAINT "fk_workid" FOREIGN KEY (workid) REFERENCES work(workid)
```

```
sales=> \d Emp
          Table "sales.emp"
   Column   |      Type      | Modifiers
-----+-----+-----
eid         | character(5)    | not null
ename       | character varying(10) |
workid      | character(3)    |
salary      | numeric(8,2)    |
phone       | character varying(11) | not null
Indexes:
    "pk_eid" PRIMARY KEY, btree (eid) TABLESPACE pg_default
    "uk_phone" UNIQUE CONSTRAINT, btree (phone) TABLESPACE pg_default
Check constraints:
    "chk_salary" CHECK (salary >= 1000::numeric)
Foreign-key constraints:
    "fk_workid" FOREIGN KEY (workid) REFERENCES work(workid)
```

(16) 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 IF EXISTS pk_eid;  
ALTER TABLE Emp DROP CONSTRAINT IF EXISTS uk_phone;  
ALTER TABLE Emp DROP CONSTRAINT IF EXISTS fk_workid;  
ALTER TABLE Emp DROP CONSTRAINT IF EXISTS chk_salary;
```

```
sales=> ALTER TABLE Emp DROP CONSTRAINT IF EXISTS pk_eid;  
ALTER TABLE  
sales=> ALTER TABLE Emp DROP CONSTRAINT IF EXISTS uk_phone;  
ALTER TABLE  
sales=> ALTER TABLE Emp DROP CONSTRAINT IF EXISTS fk_workid;  
ALTER TABLE  
sales=> ALTER TABLE Emp DROP CONSTRAINT IF EXISTS chk_salary;  
ALTER TABLE
```

(17) 删除工作表所有约束,包括主码约束。

```
ALTER TABLE Work DROP CONSTRAINT IF EXISTS pk_workid;  
ALTER TABLE Work DROP CONSTRAINT IF EXISTS  
checksalaryrange;
```

```
sales=> ALTER TABLE Work DROP CONSTRAINT IF EXISTS pk_workid;  
ALTER TABLE  
sales=> ALTER TABLE Work DROP CONSTRAINT IF EXISTS checksalaryrange;  
ALTER TABLE
```

3.实验总结

3.1 完成的工作

创建了两张表:雇员表和工作表,并在表上进行了一系列添加主键约束、外键约束、值不能高于某数的约束等等

3.2 对实验的认识

通过实验增进了对各类约束的认识,也更加熟悉添加约束的防方

法。

3.3 遇到的困难及解决方法

没遇到什么太大的困难，小困难及其解决记录

```
sales=> ALTER TABLE Work DROP CONSTRAINT IF EXISTS pk_workid;  
ALTER TABLE  
sales=> ALTER TABLE Work DROP CONSTRAINT IF EXISTS checksalaryrange;  
ALTER TABLE
```

在上

边的步骤里了。