数据库系统课程实验报告

实验名称：	数据库的完整性
实验日期：	2022/5/5
实验地点：	四号楼
提交日期：	2022/5/6

学号：	22920202202879
姓名：	陈奕培
专业年级：	软工 2020 级
学年学期：	2021-2022 学年第二学期

1.实验目的

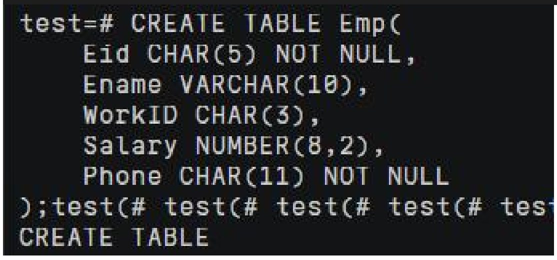
- 掌握数据库的特点（字体：华文仿宋，字号：四号，下同）

2.实验内容和步骤

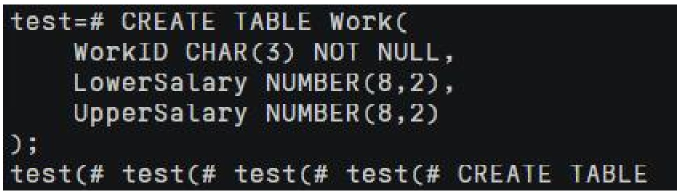
（1）创建两张表

```
CREATE TABLE Emp(
    Eid CHAR(5) NOT NULL,
    Ename VARCHAR(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)
);
```



```
test=# CREATE TABLE Emp(
    Eid CHAR(5) NOT NULL,
    Ename VARCHAR(10),
    WorkID CHAR(3),
    Salary NUMBER(8,2),
    Phone CHAR(11) NOT NULL
);test(# test(# test(# test(# tes
CREATE TABLE
```

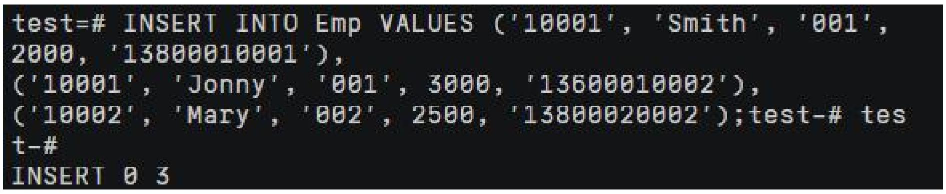


```
test=# CREATE TABLE Work(
    WorkID CHAR(3) NOT NULL,
    LowerSalary NUMBER(8,2),
    UpperSalary NUMBER(8,2)
);
test(# test(# test(# test(# CREATE TABLE
```

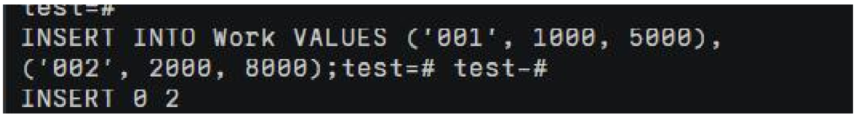
（2）插入数据

```
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);
```



```
test=# INSERT INTO Emp VALUES ('10001', 'Smith', '001',
2000, '13800010001'),
('10001', 'Jonny', '001', 3000, '13600010002'),
('10002', 'Mary', '002', 2500, '13800020002');test-# tes
t-#
INSERT 0 3
```

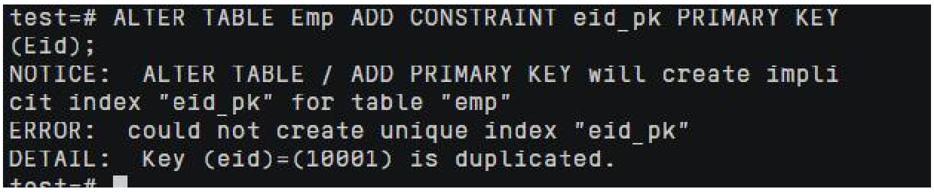


```
test=#
INSERT INTO Work VALUES ('001', 1000, 5000),
('002', 2000, 8000);test=# test-#
INSERT 0 2
```

（3） 为 Emp 添加主键

```
ALTER TABLE Emp ADD CONSTRAINT eid_pk PRIMARY KEY(Eid);
```

添加失败

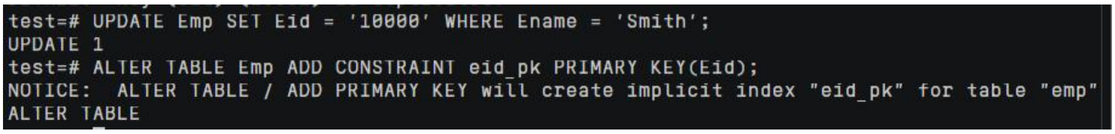


```
test=# ALTER TABLE Emp ADD CONSTRAINT eid_pk PRIMARY KEY
(Eid);
NOTICE:  ALTER TABLE / ADD PRIMARY KEY will create impli
cit index "eid_pk" for table "emp"
ERROR:  could not create unique index "eid_pk"
DETAIL:  Key (eid)=(10001) is duplicated.
test=#
```

原因是 eid=10001 重复出现了两次，违反了实体完整性约束

将其中一个 eid 改为 10000 即可，例如

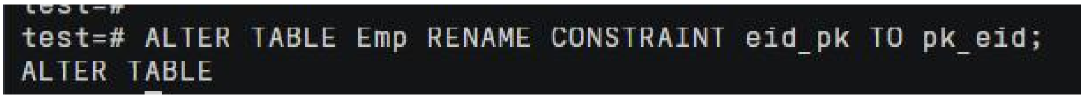
```
UPDATE Emp SET Eid = '10000' WHERE Ename = 'Smith';
```



```
test=# UPDATE Emp SET Eid = '10000' WHERE Ename = 'Smith';
UPDATE 1
test=# ALTER TABLE Emp 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） 修改约束名

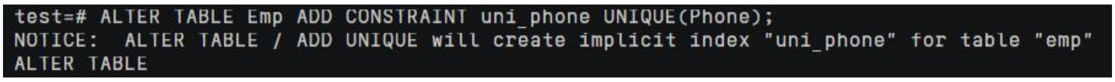
```
ALTER TABLE Emp RENAME CONSTRAINT eid_pk TO pk_eid;
```



```
test=#
test=# ALTER TABLE Emp RENAME CONSTRAINT eid_pk TO pk_eid;
ALTER TABLE
```

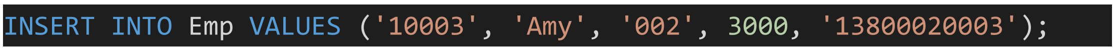
（5） 设 phone 为唯一值

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

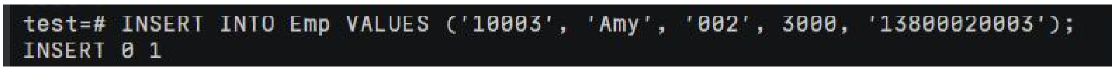


```
test=# ALTER TABLE Emp ADD CONSTRAINT uni_phone UNIQUE(Phone);
NOTICE:  ALTER TABLE / ADD UNIQUE will create implicit index "uni_phone" for table "emp"
ALTER TABLE
```

(6)给雇员表添加一条新记录(‘10003’,’Amy’,’002’, 3000,’13800020003’)



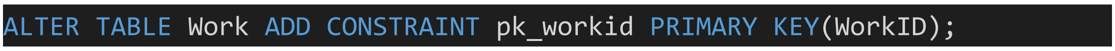
```
INSERT INTO Emp VALUES ('10003', 'Amy', '002', 3000, '13800020003');
```



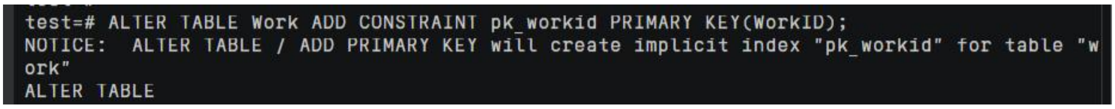
```
test=# INSERT INTO Emp VALUES ('10003', 'Amy', '002', 3000, '13800020003');
INSERT 0 1
```

插入成功，因为 id 不为空且不重复，phone 也不重复

(7) 为 Work 设置主键

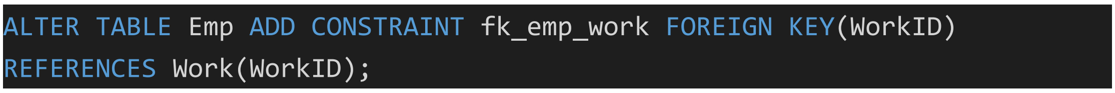


```
ALTER TABLE Work ADD CONSTRAINT pk_workid PRIMARY KEY(WorkID);
```

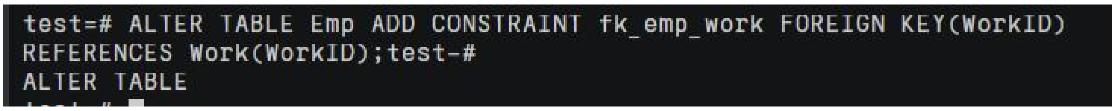


```
test=# ALTER TABLE Work ADD CONSTRAINT pk_workid PRIMARY KEY(WorkID);
NOTICE:  ALTER TABLE / ADD PRIMARY KEY will create implicit index "pk_workid" for table "w
ork"
ALTER TABLE
```

(8) 为 Emp 设置外键

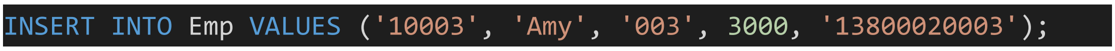


```
ALTER TABLE Emp ADD CONSTRAINT fk_emp_work FOREIGN KEY(WorkID)
REFERENCES Work(WorkID);
```

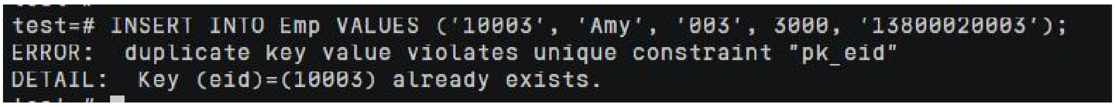


```
test=# ALTER TABLE Emp ADD CONSTRAINT fk_emp_work FOREIGN KEY(WorkID)
REFERENCES Work(WorkID);test-#
ALTER TABLE
```

(9) 给雇员表添加一条新记录(‘10003’,’Amy’, ‘003’, 3000,
‘13800020003’)

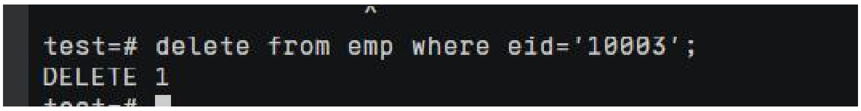


```
INSERT INTO Emp VALUES ('10003', 'Amy', '003', 3000, '13800020003');
```



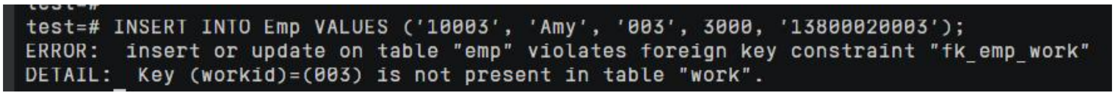
```
test=# INSERT INTO Emp VALUES ('10003', 'Amy', '003', 3000, '13800020003');
ERROR:  duplicate key value violates unique constraint "pk_eid"
DETAIL:  Key (eid)=(10003) already exists.
```

插入失败，因为 Emp 中存在 eid 为 10003 的元组



```
test=# delete from emp where eid='10003';
DELETE 1
```

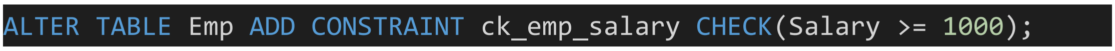
删除之前的元组后



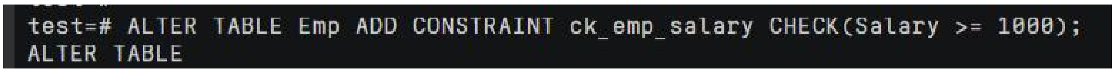
```
test=# INSERT INTO Emp VALUES ('10003', 'Amy', '003', 3000, '13800020003');
ERROR:  insert or update on table "emp" violates foreign key constraint "fk_emp_work"
DETAIL:  Key (workid)=(003) is not present in table "work".
```

任然失败，原因是 work 表中不存在 workid=003 的元组，添加的数据不满足参照完整性定义约束

（10）在雇员表中，设置雇员工资必须大于或等于 1000

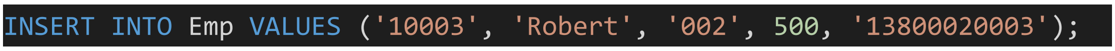


```
ALTER TABLE Emp ADD CONSTRAINT ck_emp_salary CHECK(Salary >= 1000);
```

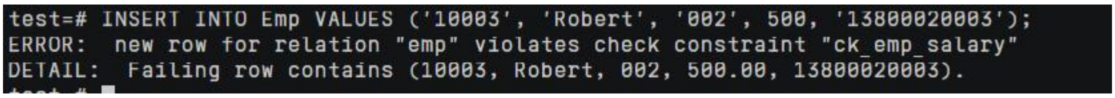


```
test=# ALTER TABLE Emp ADD CONSTRAINT ck_emp_salary CHECK(Salary >= 1000);
ALTER TABLE
```

（11）给雇员表添加一条新记录(‘10003’,’Robert’,‘002’,500,‘13800020003’)



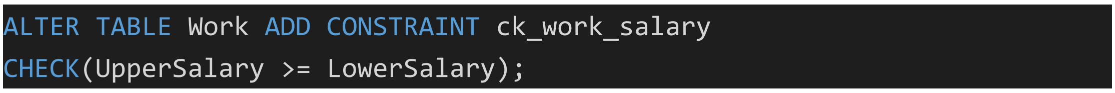
```
INSERT INTO Emp VALUES ('10003', 'Robert', '002', 500, '13800020003');
```



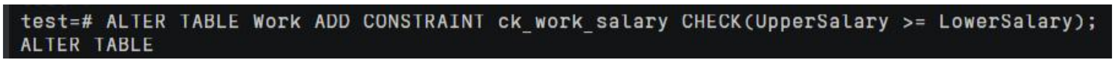
```
test=# INSERT INTO Emp VALUES ('10003', 'Robert', '002', 500, '13800020003');
ERROR:  new row for relation "emp" violates check constraint "ck_emp_salary"
DETAIL:  Failing row contains (10003, Robert, 002, 500.00, 13800020003).
```

插入失败，违反了用户定义的完整性约束

（12）在工作表中，设置其最低工资不超过最高工资

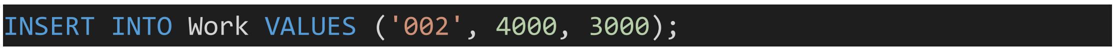


```
ALTER TABLE Work ADD CONSTRAINT ck_work_salary
CHECK(UpperSalary >= LowerSalary);
```

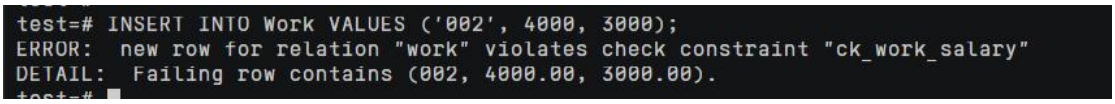


```
test=# ALTER TABLE Work ADD CONSTRAINT ck_work_salary CHECK(UpperSalary >= LowerSalary);
ALTER TABLE
```

（13）给工作表添加一条新记录(‘002’,4000,3000)



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

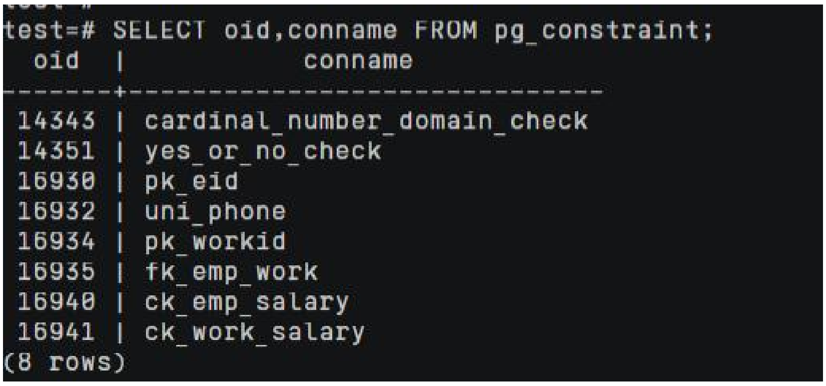


```
test=# INSERT INTO Work VALUES ('002', 4000, 3000);
ERROR:  new row for relation "work" violates check constraint "ck_work_salary"
DETAIL:  Failing row contains (002, 4000.00, 3000.00).
```

插入失败，违反了用户定义的完整性约束

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

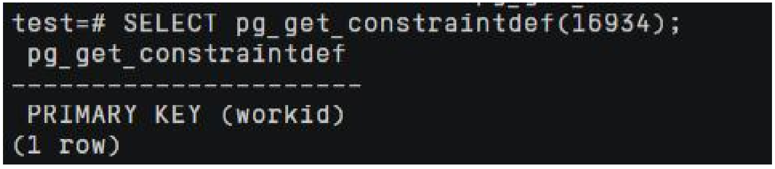
```
SELECT oid,conname FROM pg_constraint;
```



```
test=# SELECT oid,conname FROM pg_constraint;
  oid   |           conname
-------+-----------------------------
 14343 | cardinal_number_domain_check
 14351 | yes_or_no_check
 16930 | pk_eid
 16932 | uni_phone
 16934 | pk_workid
 16935 | fk_emp_work
 16940 | ck_emp_salary
 16941 | ck_work_salary
(8 rows)
```

先得到约束的 oid

```
SELECT pg_get_constraintdef(16934);
```

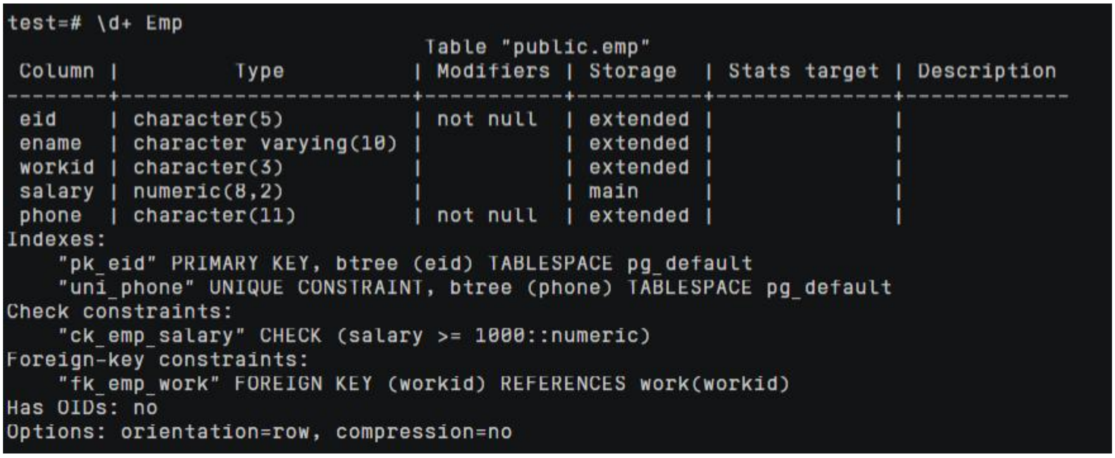


```
test=# SELECT pg_get_constraintdef(16934);
 pg_get_constraintdef
----------------------
 PRIMARY KEY (workid)
(1 row)
```

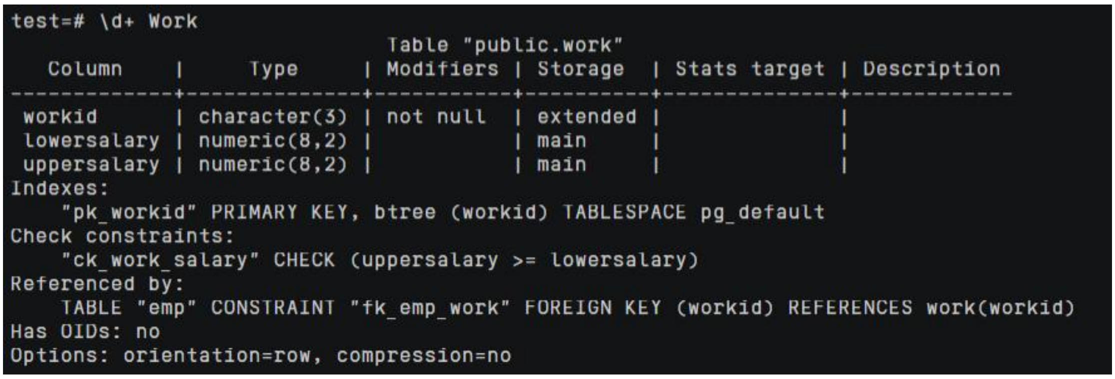
再通过 oid 去查看约束的详情

(15) 通过 gsql 命令\d+ table_name 查看改表上的约束定义

```
\d+ Emp
\d+ Work
```



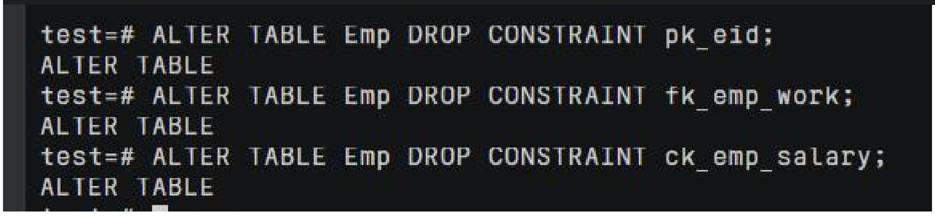
```
test=# \d+ Emp
                                 Table "public.emp"
 Column |         Type          | Modifiers | Storage  | Stats target | Description
--------+-----------------------+-----------+----------+--------------+-------------
 eid    | character(5)          | not null  | extended |              |
 ename  | character varying(10) |           | extended |              |
 workid | character(3)          |           | extended |              |
 salary | numeric(8,2)          |           | main     |              |
 phone  | character(11)         | not null  | extended |              |
Indexes:
    "pk_eid" PRIMARY KEY, btree (eid) TABLESPACE pg_default
    "uni_phone" UNIQUE CONSTRAINT, btree (phone) TABLESPACE pg_default
Check constraints:
    "ck_emp_salary" CHECK (salary >= 1000::numeric)
Foreign-key constraints:
    "fk_emp_work" FOREIGN KEY (workid) REFERENCES work(workid)
Has OIDs: no
Options: orientation=row, compression=no
```



```
test=# \d+ Work
                                  Table "public.work"
   Column    |     Type     | Modifiers | Storage  | Stats target | Description
-------------+--------------+-----------+----------+--------------+-------------
 workid      | character(3) | not null  | extended |              |
 lowersalary | numeric(8,2) |           | main     |              |
 uppersalary | numeric(8,2) |           | main     |              |
Indexes:
    "pk_workid" PRIMARY KEY, btree (workid) TABLESPACE pg_default
Check constraints:
    "ck_work_salary" CHECK (uppersalary >= lowersalary)
Referenced by:
    TABLE "emp" CONSTRAINT "fk_emp_work" FOREIGN KEY (workid) REFERENCES work(workid)
Has OIDs: no
Options: orientation=row, compression=no
```

（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

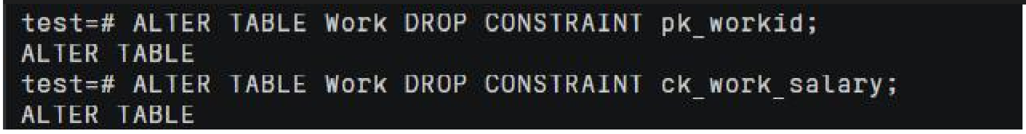
```
ALTER TABLE Emp DROP CONSTRAINT pk_eid;
ALTER TABLE Emp DROP CONSTRAINT fk_emp_work;
ALTER TABLE Emp DROP CONSTRAINT ck_emp_salary;
```



```
test=# ALTER TABLE Emp DROP CONSTRAINT pk_eid;
ALTER TABLE
test=# ALTER TABLE Emp DROP CONSTRAINT fk_emp_work;
ALTER TABLE
test=# ALTER TABLE Emp DROP CONSTRAINT ck_emp_salary;
ALTER TABLE
```

（17）删除工作表所有约束，包括主码约束

```
ALTER TABLE Work DROP CONSTRAINT pk_workid;
ALTER TABLE Work DROP CONSTRAINT ck_work_salary;
```



```
test=# ALTER TABLE Work DROP CONSTRAINT pk_workid;
ALTER TABLE
test=# ALTER TABLE Work DROP CONSTRAINT ck_work_salary;
ALTER TABLE
```

实验思考

· openGauss 实现完整性规则的机制是什么？

通过设置完整性约束来是实现完整性约束；

· 在 SQL 语句中实现完整性规则的常见约束有哪些？各自适应什么业务场景？

有三种常见约束：实体完整性约束，参照完整性约束，用户定义的完

整性约束；

实体完整性适用于需要主键，或某值需取唯一值时；

参照完整性适用于某值需要参照另外一个表中的主码的时候, 通常用于表达表之间的关系；

用户定义的完整性约束适用于用户的特定需求；

3.实验总结

3.1 完成的工作

创建两张表；

为表插入数据；

为表添加主键；

修改主键的约束名；

为表添加 unique 约束；

为表添加外键；

为表定义 check 语句；

删除表的约束；

3.2 对实验的认识

掌握了表的约束的添加方法；

修改表的约束名的方法；

当不满足完整性约束时，数据库会拒绝操作；

删除表的约束的方法；

3.3 遇到的困难及解决方法

无