数据库原理实验报告

题目： 北京工业大学社团管理系统

学号： 21071003

姓名： 高立扬

指导教师： 杜金莲

日期： 2023年11月

**评语：**

1：设计部分：

2：上机部分：

3：其它部分：

**总分：**

目录

[实验说明 4](#_Toc2151)

[1. 实验环境 4](#_Toc32276)

[数据库设计 4](#_Toc14368)

[1.选题描述 4](#_Toc32309)

[2.数据描述 4](#_Toc16124)

[3.应用描述 5](#_Toc13442)

[4.E/R图 6](#_Toc6545)

[5.关系模式 6](#_Toc4588)

[6.范式判断 7](#_Toc2298)

[7.数据表设计（取值约束，主键外键约束及其他相关约束） 7](#_Toc16228)

[实验一 创建与删除数据库 10](#_Toc21910)

[1. 实验目的 10](#_Toc544)

[2. 实验内容 10](#_Toc7380)

[2.1 创建数据库 10](#_Toc25268)

[2.2 删除数据库 12](#_Toc10829)

[3. 实验中出现的问题及分析 13](#_Toc13895)

[4. 实验中未解决的问题 13](#_Toc7865)

[5. 思考题 14](#_Toc25139)

[6. 心得体会 14](#_Toc31620)

[实验二 创建和删除基本表 14](#_Toc13766)

[1. 实验目的 14](#_Toc8204)

[2. 实验内容 15](#_Toc21258)

[2.1 创建表 15](#_Toc28111)

[2.2 修改表 16](#_Toc9909)

[2.3 删除表 19](#_Toc13077)

[3. 实验中出现的问题及分析 20](#_Toc12465)

[4. 实验中未解决的问题 23](#_Toc14295)

[5. 思考题 23](#_Toc10722)

[6. 心得体会 24](#_Toc26358)

[实验三 数据的增删改 24](#_Toc14608)

[1. 实验目的 24](#_Toc7669)

[2. 实验内容 24](#_Toc14361)

[2.1 插入数据 24](#_Toc16352)

[2.2 修改数据 26](#_Toc21476)

[2.3 删除数据 28](#_Toc19456)

[3. 实验中出现的问题及分析 29](#_Toc25763)

[4. 实验中未解决的问题 30](#_Toc32646)

[5. 思考题 30](#_Toc2405)

[6. 心得体会 30](#_Toc355)

[实验四 数据的检索 -- 第一部分 30](#_Toc10509)

[1. 实验目的 30](#_Toc20946)

[2. 实验内容 31](#_Toc19494)

[2.1 SELECT查询 31](#_Toc17266)

[2.2 聚集函数查询 31](#_Toc7723)

[2.3 DISTINCT去重 32](#_Toc12759)

[2.4 ORDER BY子句查询 32](#_Toc6716)

[2.5 使用范围条件、算术运算符、逻辑运算符查询 33](#_Toc2558)

[2.6 使用空值条件查询 34](#_Toc18011)

[2.7 字符串的模式匹配查询 35](#_Toc6920)

[2.8 GROUP BY & HAVING查询 35](#_Toc14754)

[3. 实验中出现的问题及分析 37](#_Toc6694)

[4. 实验中未解决的问题 37](#_Toc19726)

[5. 思考题 37](#_Toc23207)

[6. 心得体会 38](#_Toc5084)

[实验四 数据的检索 -- 第二部分 38](#_Toc881)

[1. 实验目的 38](#_Toc24453)

[2. 实验内容 38](#_Toc5504)

[2.1 内连接查询 38](#_Toc29915)

[2.2 派生表查询 39](#_Toc29466)

[2.3 自连接查询 40](#_Toc729)

[2.4 INTERSECT查询 41](#_Toc21974)

[2.5 UNION查询 42](#_Toc12577)

[2.6 EXCEPT查询 43](#_Toc20907)

[2.7 IN子查询 44](#_Toc2878)

[2.8 ORDER BY 子句与连接查询 45](#_Toc13726)

[2.9 EXISTS查询 45](#_Toc30501)

[2.10 ANY查询 46](#_Toc24269)

[2.11 综合查询应用 47](#_Toc6280)

[3. 实验中出现的问题及分析 48](#_Toc30955)

[4. 实验中未解决的问题 48](#_Toc20473)

[5. 思考题 48](#_Toc16223)

[6. 心得体会 48](#_Toc15343)

[实验五 创建和删除视图 49](#_Toc11434)

[1. 实验目的 49](#_Toc4906)

[2. 实验内容 49](#_Toc12741)

[2.1 创建视图 49](#_Toc11023)

[2.2 修改视图 50](#_Toc7567)

[2.3 删除视图 51](#_Toc20968)

[2.4 T-SQL操作视图 52](#_Toc23683)

[2.5 限制引用表的删除 53](#_Toc9267)

[2.6 设置视图加密 54](#_Toc15183)

[3. 实验中出现的问题及分析 55](#_Toc32102)

[4. 实验中未解决的问题 56](#_Toc27402)

[5. 思考题 56](#_Toc26578)

[6. 心得体会 57](#_Toc15376)

[实验六 创建和删除索引 57](#_Toc19588)

[1. 实验目的 57](#_Toc27857)

[2. 实验内容 57](#_Toc27482)

[2.1 设置UNIQUE约束 57](#_Toc5149)

[2.2 创建聚集索引 58](#_Toc9388)

[2.3 查看索引 58](#_Toc10920)

[2.4 重新生成索引 59](#_Toc10541)

[2.5 删除索引 59](#_Toc5599)

[3. 实验中出现的问题及分析 60](#_Toc2553)

[4. 实验中未解决的问题 60](#_Toc26186)

[5. 思考题 60](#_Toc12777)

[6. 心得体会 61](#_Toc19977)

[总结与体会 61](#_Toc23199)

#### **实验说明**

**1.实验环境**

Microsoft SQL Server Enterprise Evaluation (64-bit) 16.0.1105.1 openEuler20.03 Red Hat （64-bit）

#### **数据库设计**

**1.选题描述**

每年新生入学季，为了进一步丰富学生的校园生活、繁荣北工大的校园文化、践行北工大的校训“不息为体，日新为道”精神，校团委社团管理部会主持组织全体学生，分别在北京工业大学通州校区和平乐园校区开展“百团嘉年华”活动，也就是我们工大学子每年秋季的热门话题“百团大战”。2023年9月，“百团大战”成功在上述两个校区举办，会场上除了老牌社团，还可以看到新成立的社团，新老社团共同使工大“百团”迸发着无尽的激情。工大“百团”种类丰富，兼顾了思想、知识、艺术等领域，多种多样的社团促进工大学子全方面发展，助力工大青年德智体美劳全方位发展。故我以“北京工业大学社团管理系统”为选题，此系统便于工大社团的信息管理，便于即将加入社团的学生查看社团信息，助力工大社团的更好发展，助力工大学子体验到更加丰富多彩的大学生活。

**2.数据描述**

首先声明**实体集**如下：

工大学子BJUTer：名Sname、学号Sid、年级Sgrade、学部Sxb、参与数量Sp和在社团内的职称level。学号是键；

社团club：社团名称Cname、社团状态Cstate、创建时间Ccreate和社团的类型Ctype。社团名称是键；

部门department：部名Dname和职能Dfunc。部名是键；

教室teacher：姓名Tname、学部Txb、参与数量Tp和工号Tid。工号是键；

办公室office：办公室所属学部Oxb、所属楼宇Ob和门牌号Obr。全键；

活动activity：活动地点Aplace，活动时间Atime和活动名称info。活动名称是键。

接下来是数据描述：

1. 工大学子可以参加多个社团，社团可以容纳多个工大学子，一名工大学子最多同时参加十个社团（多对多）；
2. 一名工大学子仅可以作为一个社团的社长，一个社团仅能有一位社长（一对一）；
3. 一个社团包含了多个部门（一对多）
4. 一名工大学子仅可作为一个部门的部长，一个部门仅可以有一名部长（一对一）；
5. 多名教师可以指导同一个社团，一名教师可以指导多个社团，一名老师最多同时指导十个社团（多对多）；
6. 一个社团可以有一个或以上个专属办公室（一对多）；
7. 一个社团可以举办很多活动，一个活动可以由多个社团共同举办（多对多）。

由此可得**联系集**如下：

1. 部长secretary：无属性；
2. 社长leader：记录isL，同时也是键；
3. 包含includes：无属性；
4. 参加joinin：无属性；
5. 指导guide：无属性；
6. 拥有have：无属性；
7. 举办hold：无属性。

最后，对一些属性进行说明：

1. 对于工大学子的职称，如果没参加社团则为NULL，参加的话，限制为“部长”，“社长”和“干事”，一个学生若参加多个社团，则职称按照最高的那个来进行赋值，学生退出所有社团后，职位仍会保留，作为学生曾参加过社团的依据；
2. 社团类型Ctype暂定为：艺术类、体育类、创新类、文化类和思想类，**在下文7.中用λ来表示上述字符串**，社团管理人员可以后续根据新社团的加入，添加新的社团类型；
3. 各个社团的部门会根据只能而变化其名称，这里暂定为“XX社团”为该社团所有部门的前缀，后缀分别可以选择组织部、策划部、办公部和实践部，**在下文7.中用γ来表示上述字符串**；
4. 在一些活动中，经常需要指派社团社长和对应部门部长来组织策划，因此需要经常查询部长和社长，因此应当针对部长和社长进行关系模式的建立；
5. 社团状态Cstate指的是社团是否已注册或已注销，注销后应当移出社团的所有成员，但是不应当删除社团信息和其已举办过的活动，这便于查看社团信息的学生可以重启已注销的社团，或以注销的社团为基础创立同类型的社团。

**3.应用描述**

根据以上实体集、数据描述、联系集和说明，来确定选题的相关查询、更新等操作如下：

1. 查询所有社团信息；
2. 查询所有社团社长信息；
3. 查询社团各部长信息；
4. 查询社团指导教师信息；
5. 查询社团办公室地点；
6. 工大学子涉及到职位任免和进退社团时，更新相关关系模式；
7. 每年开学季/毕业季的时候，对工大学子名单进行增删，并更新相关关系模式；
8. 社团举办活动后，更新相关关系模式；
9. 教师涉及到入职离职和指导社团变更时，更新相关关系模式；
10. 社团的办公室改变时，更新相关关系模式；
11. 社团注册或注销，更新相关关系模式；

**4.E/R图**

综上，作E/R图如下图1

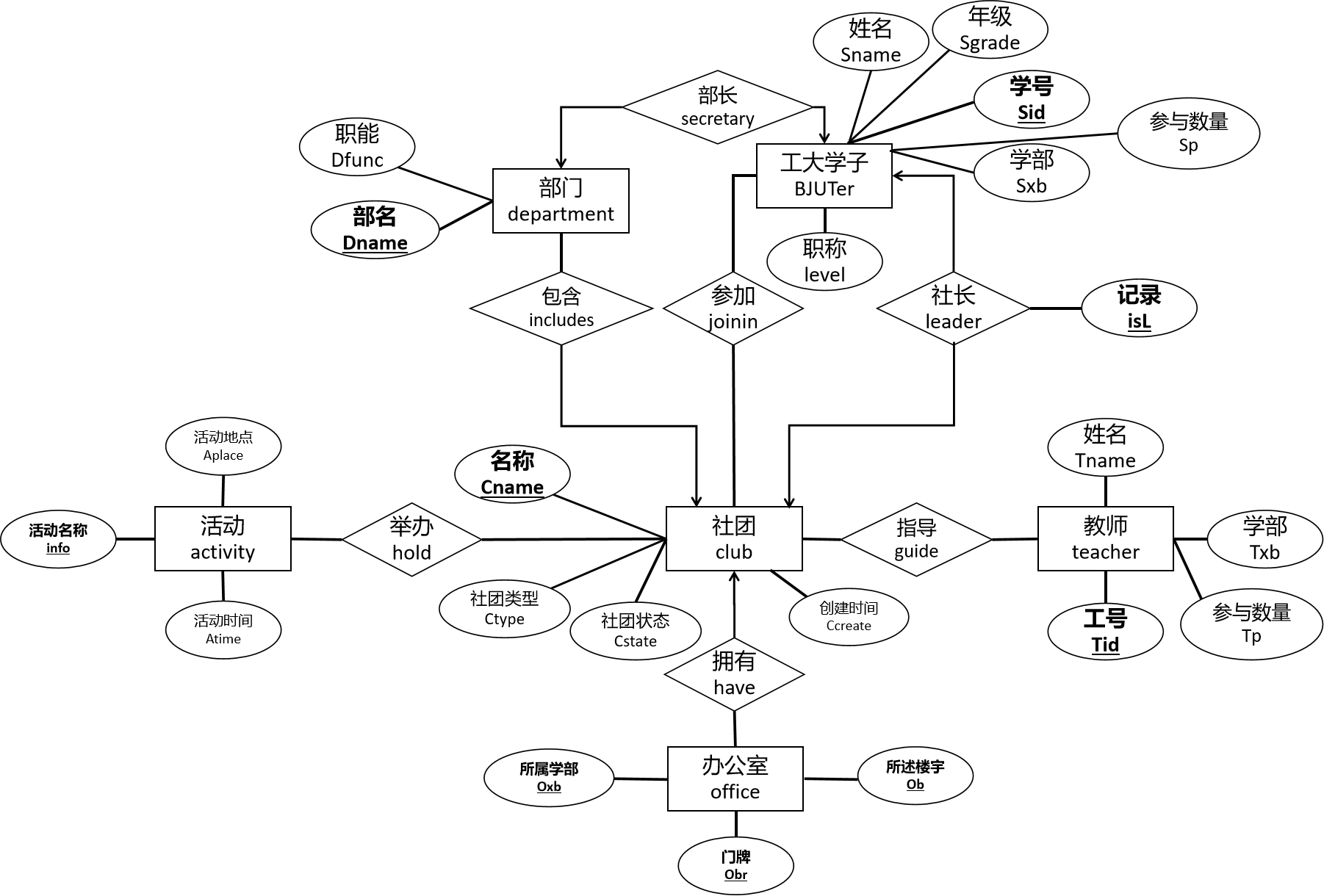


图1 E/R图

**5.关系模式**

综上所述，最终得到关系模式如下：

BJUTer(**Sid**, Sname, level, Sgrade, Sxb, Sp)

secretary(**Sid**, Dname)

department(**Dname**, Dfunc)

leader(**Sid**, Cname, isL)

joinin(**Sid**, **Cname**)

includes(Dname, **Cname**)

club(**Cname**, Ctype, Cstate, Ccreate)

guide(**Cname**, **Tid**)

teacher(**Tid**, Tname, Txb, Tp)

office(**Oxb,** **Ob,** **Obr**)

have(**Oxb,Ob,Obr**, Cname)

activity(**info**, Atime ,Aplace)

hold(**info**, **Cname**)

**6.范式判断**

joinin,office,hold,guide是全键，必然满足BCNF；

BJUTer的非主属性，即学生的姓名、职位、参与社团数量、学部和年级，与对应关系模式的键没有任何依赖关系，因此满足BCNF；

teacher同上，人名、社团名和参与指导社团数量与键不存在任何依赖关系，因此满足BCNF；

club的非主属性是Ctype和Cstate，一个社团的类型和状态，与其名称不存在任何依赖关系，因此满足BCNF；

activity的非主属性是Atime和Aplace，一个活动的时间和地点，与活动名称不存在任何依赖关系，因此满足BCNF；

department同理，不再赘述，满足BCNF；

对于secretary,leader,includes,have，它们是联系集，内部属性均来源于联系的双方。以secretary为例，主属性Sid与非主属性Dname，完全不存在任何依赖关系，因此满足BCNF。经过验证，其余三个联系集也都不存在相互依赖的关系，故满足BCNF

综上所述，本数据库设计合理，转换出的关系模式完全符合BCNF。

**7.数据表设计（取值约束，主键外键约束及其他相关约束）**

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| BJUTer | | | | | |
| 属性 | 属性类型 | 取值约束 | 相关约束 | 主键 | 外键 |
| Sid | CHAR(M) | M=8 | PRIMARY KEY | √ |  |
| Sname | VARCHAR(M) | 0<=M<=30 | UNIQUE |  |  |
| level | VARCHAR(M) | M=’部长’,‘社长’，’干事’ | default(NULL) |  |  |
| Sgrade | VARCHAR(M) | M=’大一’，‘大二’，‘大三’，‘大四’，‘大五（五年制）’ | - |  |  |
| Sxb | VARCHAR(M) | M=北工大的各个学部 | - |  |  |
| Sp | TINYINT M | 0≤M≤10 | default(0) |  |  |

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| secretary | | | | | |
| 属性 | 属性类型 | 取值约束 | 相关约束 | 主键 | 外键 |
| Sid | CHAR(M) | M=8 | PRIMARY KEY | √ | √ |
| Dname | VARCHAR(M) | M=γ | PRIMARY KEY |  | √ |

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| department | | | | | |
| 属性 | 属性类型 | 取值约束 | 相关约束 | 主键 | 外键 |
| Dname | VARCHAR(M) | M=γ | PRIMARY KEY | √ |  |
| Dfunc | VARCHAR(M) | 0≤M≤100 | - |  |  |

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| leader | | | | | |
| 属性 | 属性类型 | 取值约束 | 相关约束 | 主键 | 外键 |
| Sid | CHAR(M) | M=8 | PRIMARY KEY | √ | √ |
| Cname | VARCHAR(M) | 0<=M<=100 | PRIMARY KEY |  | √ |
| isL | BIT | - | default(0) |  |  |

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| joinin | | | | | |
| 属性 | 属性类型 | 取值约束 | 相关约束 | 主键 | 外键 |
| Sid | CHAR(M) | M=8 | PRIMARY KEY | √ | √ |
| Cname | VARCHAR(M) | 0<=M<=100 | PRIMARY KEY | √ | √ |

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| includes | | | | | |
| 属性 | 属性类型 | 取值约束 | 相关约束 | 主键 | 外键 |
| Dname | VARCHAR(M) | M=λ | PRIMARY KEY |  | √ |
| Cname | VARCHAR(M) | 0<=M<=100 | PRIMARY KEY | √ | √ |

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| club | | | | | |
| 属性 | 属性类型 | 取值约束 | 相关约束 | 主键 | 外键 |
| Cname | VARCHAR(M) | 0<=M<=100 | PRIMARY KEY | √ |  |
| Ctype | VARCHAR(M) | M=λ | - |  |  |
| Cstate | VARCHAR(M) | M=‘已注册’，‘已注销’ | - |  |  |
| Ccreate | DATETIME | 1960-01-01 00:00:00 <= M <= 2999-12-31 23:59:59 | - |  |  |

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| guide | | | | | |
| 属性 | 属性类型 | 取值约束 | 相关约束 | 主键 | 外键 |
| Cname | VARCHAR(M) | 0<=M<=100 | PRIMARY KEY | √ | √ |
| Tid | CHAR(M) | M=5 | PRIMARY KEY | √ | √ |

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| teacher | | | | | |
| 属性 | 属性类型 | 取值约束 | 相关约束 | 主键 | 外键 |
| Tid | CHAR(M) | M=5 | PRIMARY KEY | √ |  |
| Tname | VARCHAR(M) | 0<=M<=60 | - |  |  |
| Txb | VARCHAR(M) | M=北工大的各个学部 | - |  |  |
| Tp | TINYINT M | 0≤M≤10 | default(0) |  |  |

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| office | | | | | |
| 属性 | 属性类型 | 取值约束 | 相关约束 | 主键 | 外键 |
| Oxb | VARCHAR(M) | M=北工大的各个学部 | PRIMARY KEY | √ |  |
| Ob | VARCHAR(M) | 0≤M≤30 | PRIMARY KEY | √ |  |
| Obr | VARCHAR(M) | 0≤M≤12 | PRIMARY KEY | √ |  |

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| have | | | | | |
| 属性 | 属性类型 | 取值约束 | 相关约束 | 主键 | 外键 |
| Oxb | VARCHAR(M) | M=北工大的各个学部 | PRIMARY KEY | √ | √ |
| Cname | VARCHAR(M) | 0<=M<=100 | - |  | √ |
| Ob | VARCHAR(M) | 0≤M≤30 | PRIMARY KEY | √ | √ |
| Obr | VARCHAR(M) | 0≤M≤12 | PRIMARY KEY | √ | √ |

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| activity | | | | | |
| 属性 | 属性类型 | 取值约束 | 相关约束 | 主键 | 外键 |
| info | VARCHAR(M) | 0<=M<=60 | PRIMARY KEY | √ |  |
| Atime | DATETIME | 1960-01-01 00:00:00 <= M <= 2999-12-31 23:59:59 | - |  |  |
| Aplace | VARCHAR | 0<=M<=60 | - |  |  |

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| hold | | | | | |
| 属性 | 属性类型 | 取值约束 | 相关约束 | 主键 | 外键 |
| info | VARCHAR(M) | 0<=M<=60 | PRIMARY KEY | √ | √ |
| Cname | VARCHAR(M) | 0<=M<=100 | PRIMARY KEY | √ | √ |

#### 实验一 创建与删除数据库

## 实验目的

本实验要求使用这两种方法SQL语句创建和删除数据库，实验目的在于：

1）学习使用SQL语句建立与管理数据库。

3）学会SQL语句的排错技术。

4）了解数据文件、日志文件等相关概念。

5）建立案例数据库以及自己设计的数据库，为以后的实验做准备。

6）对常见错误操作，进行测试，加深对数据库管理相关语句以及操作的理解。

## 实验内容

2.1 创建数据库

（1）在SQL server中创建数据库

如下图1.1所示，打开SSMS之后，还没有创建数据库的状态

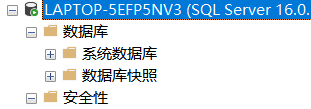


图1.1 初始状态

按照实验指导书的要求，先用USE选择master数据库为当前的数据库，然后按照实验教材和指导书，用T-SQL语句对数据库进行创建。如图1.2所示为代码，如图1.3所示为建立好ClubSystem数据库的查询结果，分别通过SSMS和T-SQL语句来进行查询

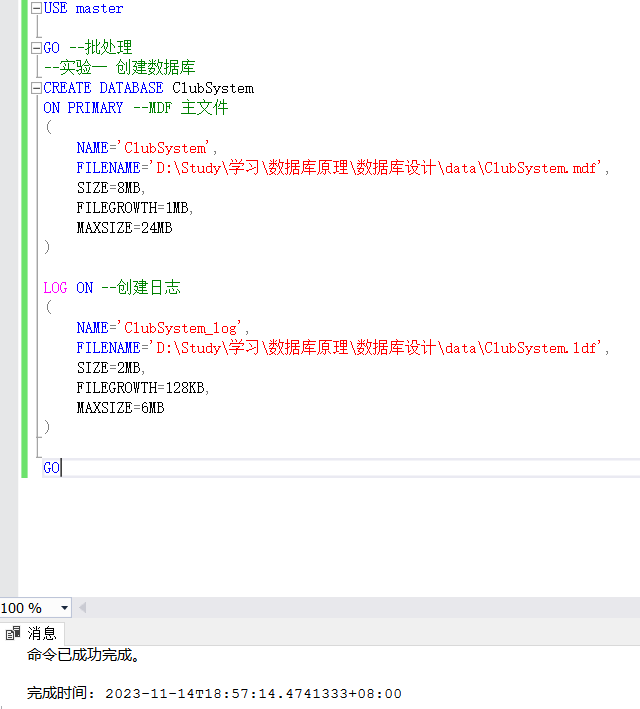


图1.2 脚本

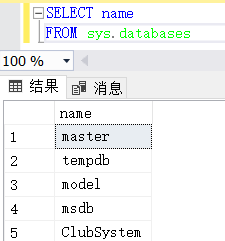
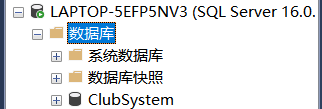


图1.3 数据库成功建立

1. 在openGauss中创建数据库

在openGauss中，首先进入omm用户，然后启动openGauss数据库后进入数据库，指令如下

su - omm

gs\_ctl start -D /data/opengauss3.0/data/db

gsql -d postgres -p 15400 -r

进入数据库后，首先用\l指令查看当前存在哪些数据库，如图1.4

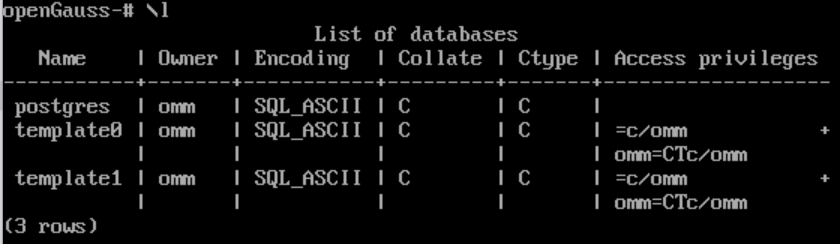


图1.4 数据库初始状态

接着，用CREATE DATABASE ClubSystem;来创建数据库，可以看见数据库创建成功，如下图1.5所示。这里可以看到，创建好的数据库为全小写，通过查阅资料发现openGauss中的数据库命名均为小写字母。在后续实验中为了代码简便，全写为大写“CLUBSYSTEM”

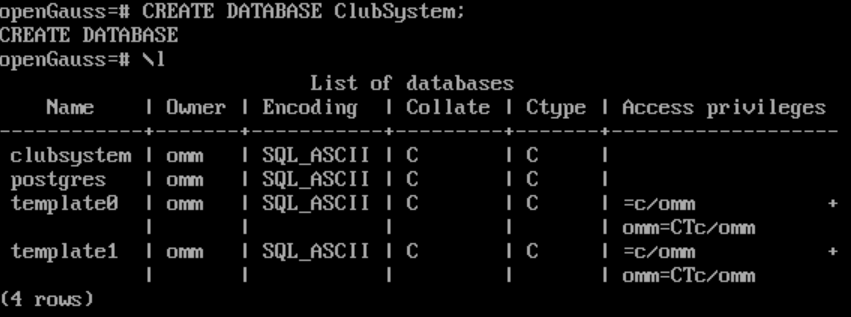


图1.5 创建成功

2.2 删除数据库

（1）在SQL server中删除数据库

在SQL server中，采用T-SQL语言来进行数据库删除，语句如下图1.6

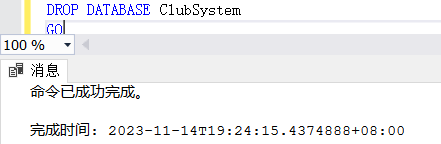


图1.6 删除数据库语句

通过2.1中的两种方式进行查询，如图1.7所示，数据库成功被删除，删除后重新创建数据库，方便后续实验的进行

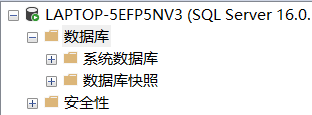
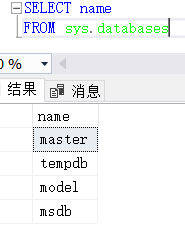


图1.7 删除后数据库的查询

（2）在openGauss中删除数据库

在openGauss中执行语句DROP DATABASE CLUBSYSTEM;后，\l指令查询数据库，发现ClubSystem被正确地删除如图1.8所示

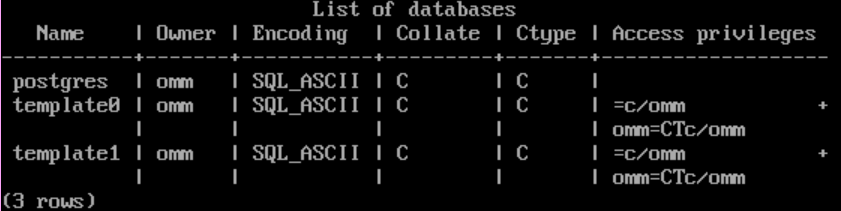


图1.8 删除后数据库的查询

## **实验中出现的问题及分析**

1. 在openGauss中第一次创建数据库的时候，发现并没有弹出任何成功或者报错的信息，如下图1.9所示，经过查询资料我了解到，这是因为我没有输入分号导致的，因此我牢记了每次输入完指令需要加分号，并在报告中的指令描述结尾也都添加了分号。

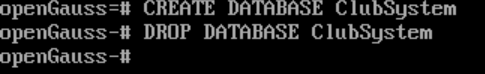


图1.9 错误地输入指令

1. 如上图1.9所示，我输入的数据库一直都是ClubSystem，但是如图1.5所示，在openGauss中，所有的数据库命名一律为小写字母，因此我在后续编程中注意到了此问题。

## **实验中未解决的问题**

本实验暂无未解决的问题

## **思考题**

Q1：数据库文件有哪些增长方式？

A1：两种。一种是按照固定长度增长，比如本实验用到的FILEGROWTH=1MB；另一种是按照百分比增长，比如FILEGROWTH=10%

Q2：日志文件的作用是什么？

A2：①能记录物理数据页面的修改的信息；②能将数据从逻辑上恢复至事务之前的状态；③能以二进制文件的形式记录了数据库中的操作；④能记录错误的相关信息；⑤能从主服务器中二进制文件取的事件等等

## **心得体会**

通过本次实验，我掌握了创建与删除数据库的操作，并学会了在SQL server和openGauss中快速查询所有的数据库信息。通过在openGauss中的操作，我注意到了每条指令结束后必须写分号，避免了后续进行复杂操作时因不写分号导致的隐患。同时我还得知了openGauss数据库的命名是全小写字母，这避免了我未来进行操作时，因大小写切换而浪费时间

在学长的报告中，我看到了GO批处理指令，虽然我也用到了我的代码中，但是我对GO指令的理解不能只停留在抄。通过自主查阅资料，我了解到了GO类似于MATLAB中的%%，将脚本分成了一块一块，每一块的执行不会影响其他块，并且还可以进行块的循环执行等

#### 实验二 创建和删除基本表

##### 实验目的

本实验的学习目标在于熟练掌握数据库基本表的创建、修改和删除的方法，具体实验目的如下：

1）学会使用SQL语句创建、修改和删除表。

2）学会使用SQL语句设置常用的数据完整性约束，含主键约束、外键约束、空值约束、UNIQUE约束、默认值以及CHECK约束等。

3）学会使用系统存储过程查看基本表信息。

4）熟悉SQL的常用数据类型。

5）理解相关概念：基本表与三级结构、实体完整性、参照完整性、用户定义完整性、主键、外键、空值、默认值等。

6）建立案例数据库以及自己设计的数据库的相关基本表，为后面的实验做准备。

7）测试各种异常、错误情况，加深对表管理操作以及相关知识点的理解。

##### **实验内容**

2.1 创建表

（1）在SQL server中创建表

根据自己的数据库设计，编写脚本来进行表的创建，首先先建其中一个表来测试脚本的正确书写，代码如下图2.1所示

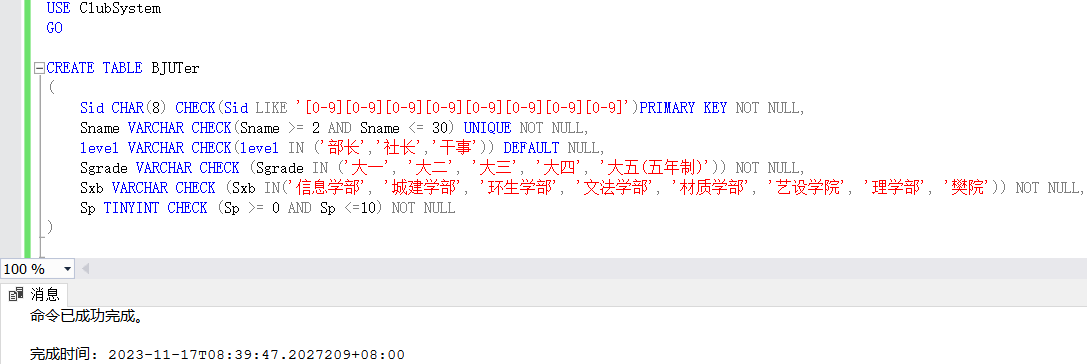


图2.1 示例代码

可以看到，表被成功创建了，如图2.2所示。下面对所有表进行创建，结果如图2.3所示，全部创建成功

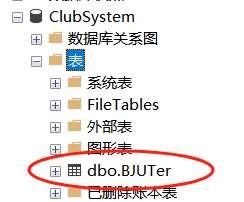
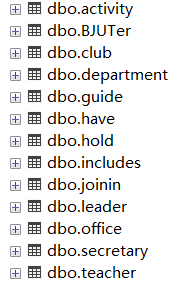
 

图2.2 成功创建单个表 图2.3 全部创建

（2）在openGauss中创建表

在命令行进行如此多的输入，显然是不现实的，于是通过查阅资料，我打算用Navicat软件来连接openGauss数据库，并进行脚本编写

要想通过Navicat来连接openGauss，首先在openGauss中创建新的用户，因为以omm用户身份是无法访问的，因此需要先在数据库内进行用户创建，并且进行授权，否则无法访问，同时，由于实验一中以omm身份创建了CLUBSYSTEM，因此还需要对新用户进行授权，还需要对建表权限进行授权，如图2.4所示，用户dbtest已经被授权clubsystem数据库的权限

CREATE USER dbtest IDENTIFIED BY ‘密码’;

GRANT CREATE ON DATABASE CLUBSYSTEM TO dbtest;

grant all on schema public to dbtest

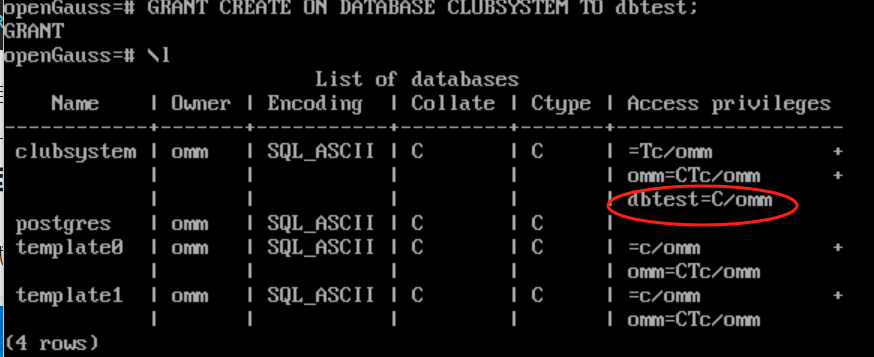


图2.4 授权完毕

接下来便是写脚本进行表的创建了，openGauss中没有datatime数据类型，因此用字符串类型进行替换；teacher表中的Tid会与数据库关键字冲突，因此改名为Tids。其余脚本代码均与SQL server建表代码相同，不再赘述，表全部创建好如图2.5所示

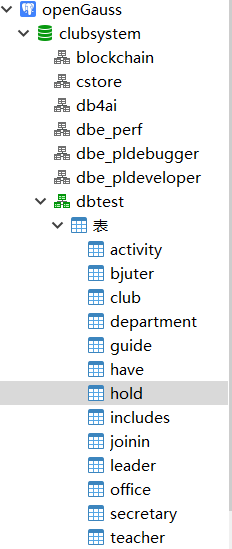


图2.5 建表完毕

2.2 修改表

（1）在SQL server中修改表

下面将会用ALTER来进行修改表的操作，具体来说，会挑选一个表，再其之中创建一个新的列，再修改列的类型，最后删除列，以完成实验要求。

如下图2.6所示，在修改之前，对BJUTer表的内容进行展示



图2.6 before alter add

下面通过图2.7所示代码，对BJUTer表进行列“BTEST”添加，如图2.8，添加成功

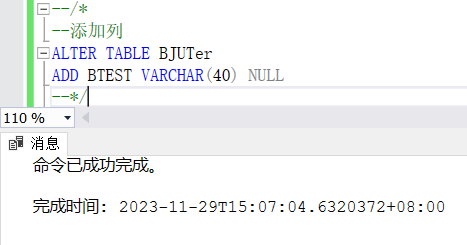


图2.7 添加列

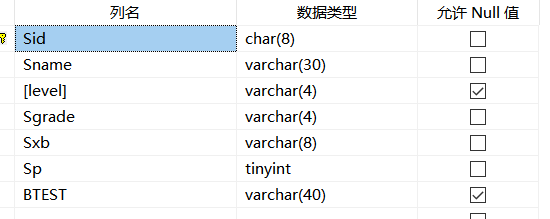


图2.8 after alter

下面通过图2.9所示代码，对BTEST列进行类型修改，修改结果如图2.10

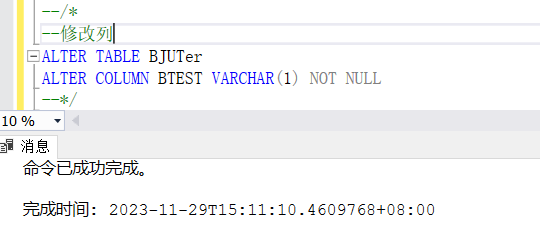


图2.9 修改列

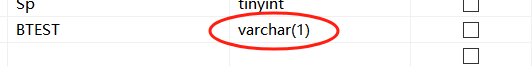


图2.10 修改完毕

最后进行列的删除，代码如图2.11所示，结果如图2.12所示

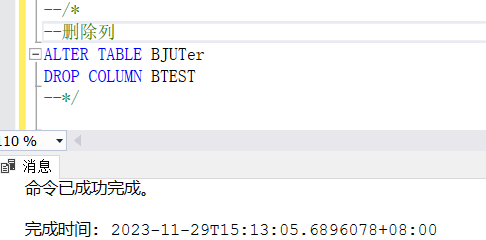


图2.11 删除列

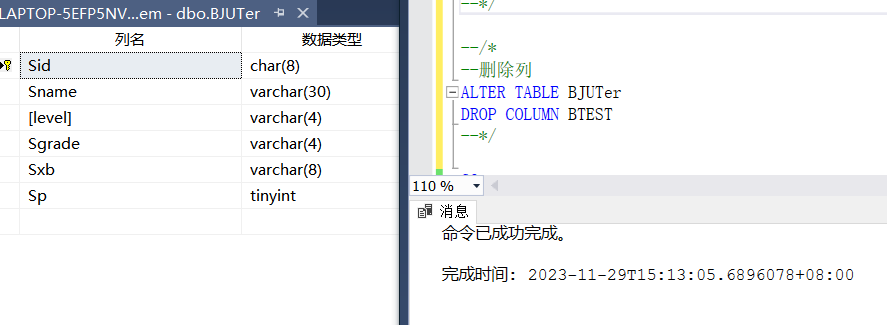


图2.12 删除成功

（2）在openGauss中修改表

openGauss中修改表的脚本代码和在SQL server中的代码相似，下面展示代码和结果，不同之处会特殊说明

创建列代码和结果如图2.13-2.15所示

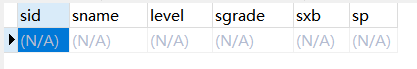


图2.13 before add

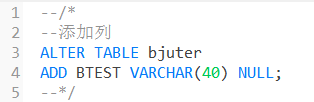


图2.14 添加列

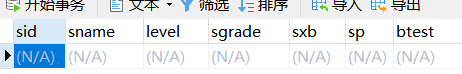


图2.15 after add

修改列代码和结果如图2.16-2.18所示，这里的代码和SQL server就有些许不同了，我参考了文章https://mac.8miu.com/read-467305.html，成功编写了能完成预期目的的代码

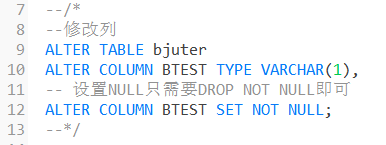


图2.16 修改

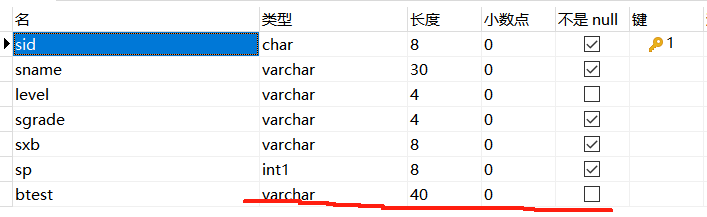


图2.17 before alter



图2.18 after alter

删除列代码和结果如图2.19所示

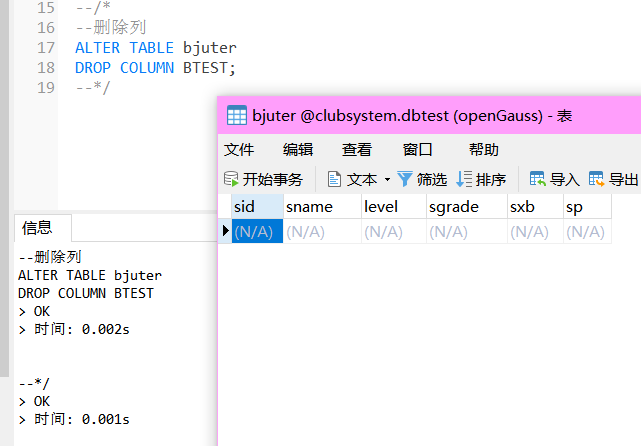


图2.19 删除成功

2.3 删除表

（1）在SQL server中删除表

如图2.20所示，表成功被删除

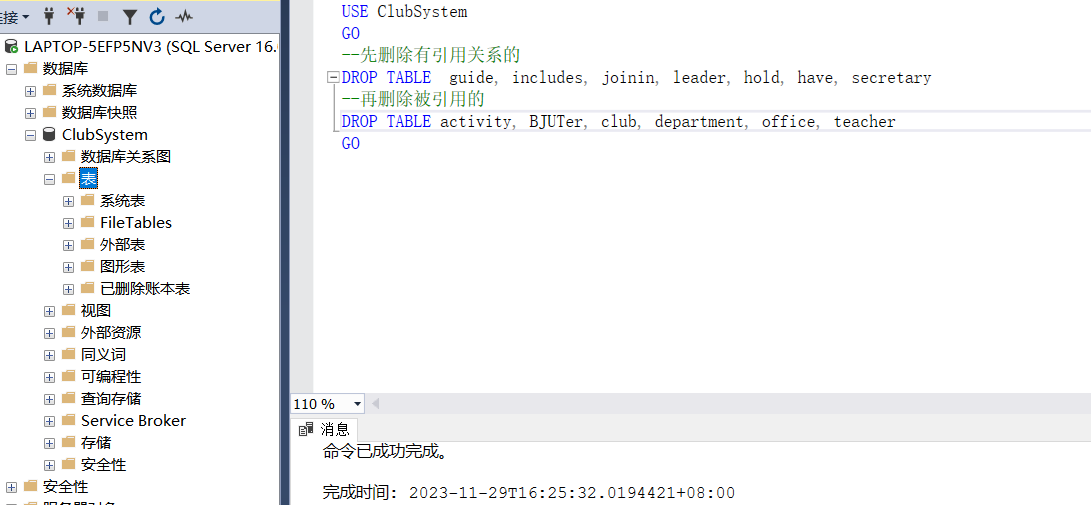


图2.20 删除成功

（2）在openGauss中修改表

如图2.21所示，表成功被删除

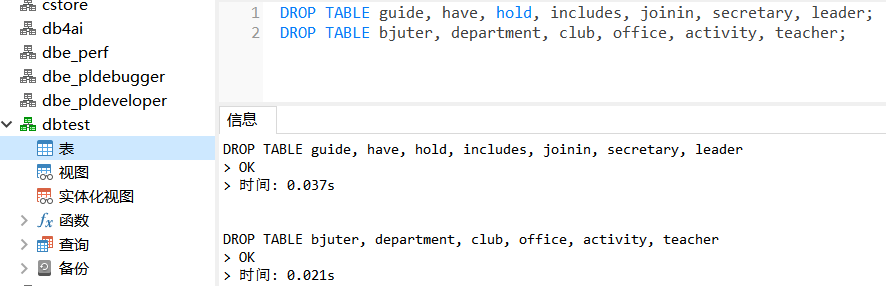


图2.21 删除成功

##### **实验中出现的问题及分析**

1. 在Navicat中以omm用户组访问openGauss失败问题

如图2.20-21所示，omm用户访问失败。解决办法是以omm用户在数据库中新建用户，然后参考下面贴出的文章来进行配置修改后，连接成功，如图2.22所示

https://blog.csdn.net/reloaded\_hjy/article/details/128852986



图2.20 omm访问

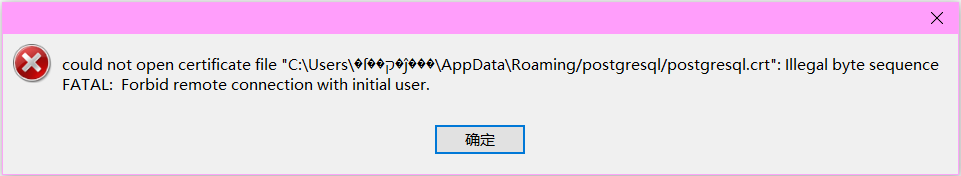


图2.21 omm无法访问



图2.22 dbtest用户可以访问

1. 创建的用户无法创建数据库

如图2.23，这是权限问题导致的，在openGauss数据库下，通过\c - omm进入超级用户，然后输入指令ALTER ROLE 用户 CREATEROLE CREATEDB即可赋予用户超级权限“create role”和“create db”，如图2.24，权限赋予成功

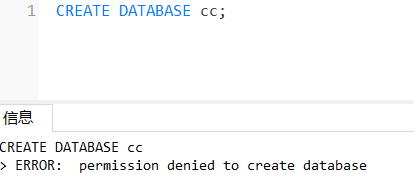


图2.23 权限不足，无法创建数据库

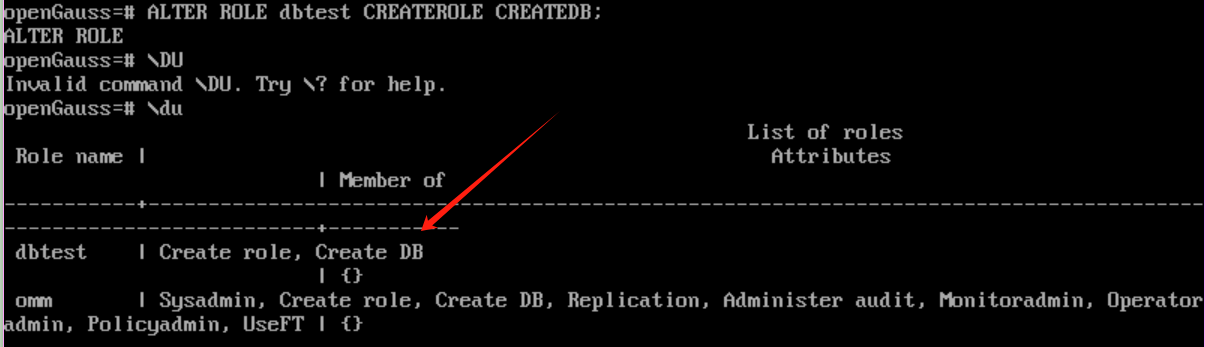


图2.24 权限添加

1. 用户获取数据库访问权限之后无法创建表

如图2.25所示，解决方法是输入grant all on schema public to 用户，赋予用户创建表的权限，结果如图2.26所示

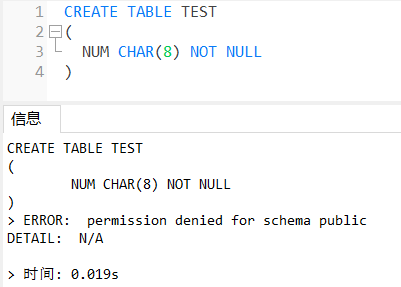


图2.25 权限不足

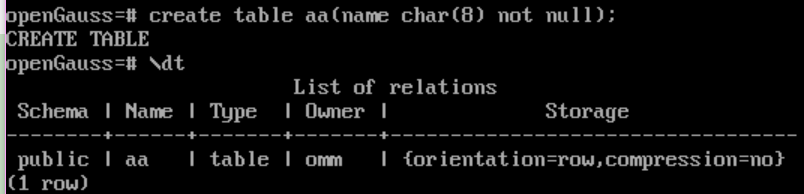


图2.26 添加权限

1. 未在理想位置建表

由于不熟悉openGauss的操作，导致没有切换到CLUBSYSTEM数据库，一直是在postgres数据库进行操作。因此需要\c clubsystem来进行数据库的切换；切换过后，用CREATE TABLE ...来进行建表，发现表建在了名为public的schema里，并且在navicat里没有发现名为dbtest的schema，如图2.27所示。解决方法是\c - dbtest切换到访问用户之后，用CREATE SCHEMA dbtest;来进行模式创建，此时再建表，就可以看见表在正确的schema内建立了，如图2.28。值得一提的是，在建立schema之前，先要按照（3）的步骤给dbtest授权

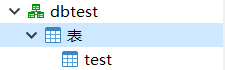
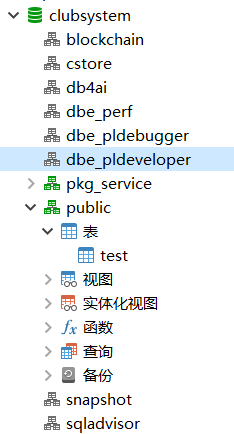


图2.27-2.28 建表得挑对地方

1. 建表错误

如图2.29，在创建外键的时候遇见了如下错误，这是因为REFERENCE的键不是UNIQUE，将被引用的表里对应的键设置为UNIQUE即可

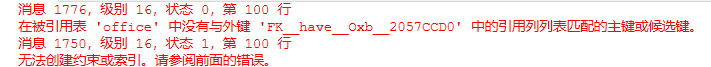


图2.29 建表错误

1. 删除错误

在drop表的时候，需要注意drop的顺序，比如A表中有外键K，K引用自B表，那么就必须先drop A才能drop B，不能先drop B

##### **实验中未解决的问题**

本实验暂无未解决的问题

##### 思考题

Q1：什么叫做外键？外键的作用是什么？

A1：外键也称为外码或外部关键字。是用于建立和加强两个表数据之间的链接的一列或多列，通过将保存表中主键值的一列或多列添加到另一个表中，可创建两个表之间的链接，这个列就成为第二个表的外键

##### 心得体会

通过本次实验，我掌握了创建和删除表的操作，还学习到了T-SQL中定义数据类型、长度、空值否、默认值、主键约束、外键约束、CHECK约束的技巧。在写openGauss相关脚本的时候，遇见了与SQL server不同的语法，于是通过查阅资料，顺利完成了任务，也体会到了两种数据库交汇贯通之处

#### 实验三 数据的增删改

##### 实验目的

有关数据库中表的更新操作的实验，主要目的是：

1）学会使用SQL语句进行数据的增删改。

2）掌握数据增删改对数据约束的影响，深入理解主键约束、外键约束、check约束以及空值、默认值等相关概念。

3）熟练掌握各种数据类型的使用。

4）对于案例数据库以及自己设计的数据库中的基本表，插入数据，作为后面查询实验的基础

##### 实验内容

2.1 插入数据

（1）在SQL server中插入数据

SQL server中插入语句为INSERT INTO...。插入操作可以用两种方式来实现，第一种是VALUES子句；第二种是SELECT子句，多条插入需要用SELECT...UNOIN...

如图3.1所示，该图为VALUES单条插入

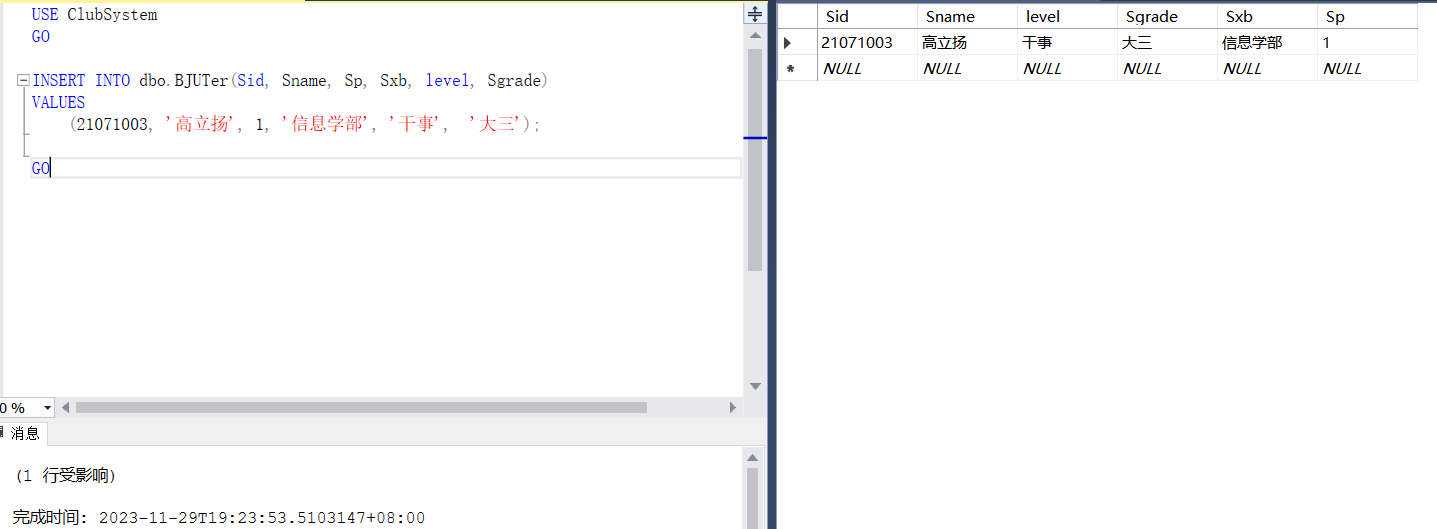


图3.1 VALUES单条插入

如图3.2所示，该图为VALUES多条插入

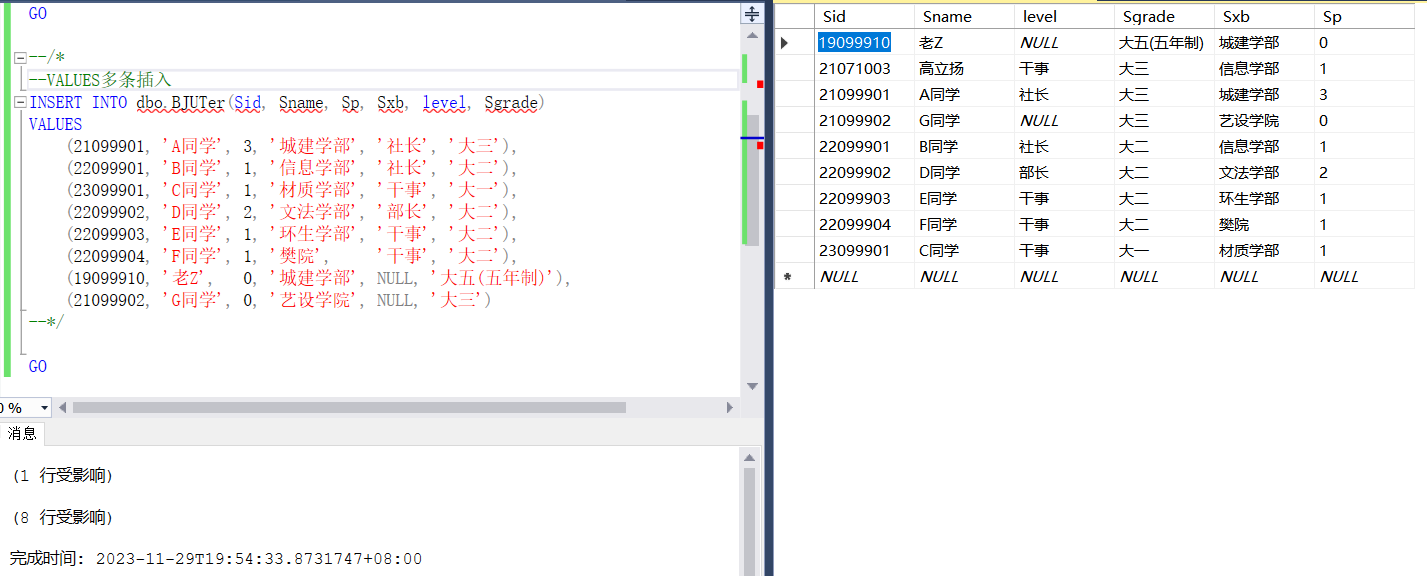


图3.2 VALUES多条插入

如图3.3所示，该图为SELECT多条插入

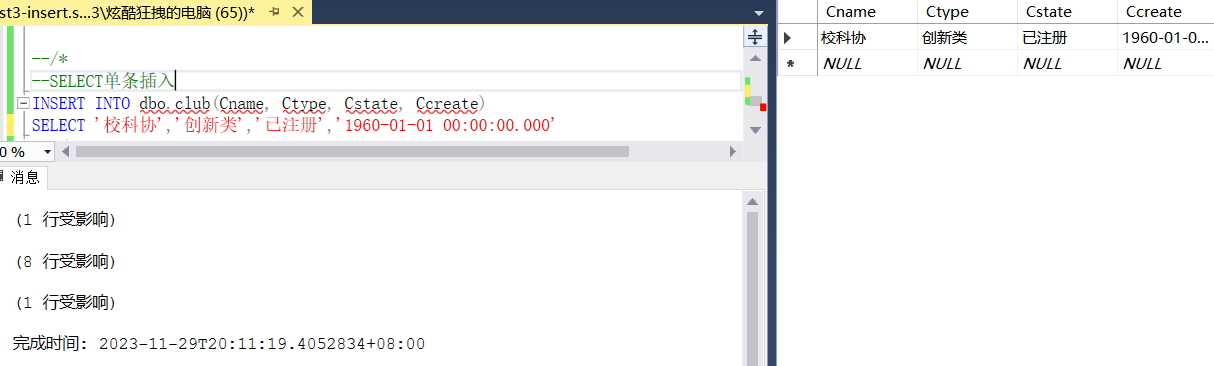


图3.3 SELECT单条插入

如图3.4所示，该图为SELECT多条插入

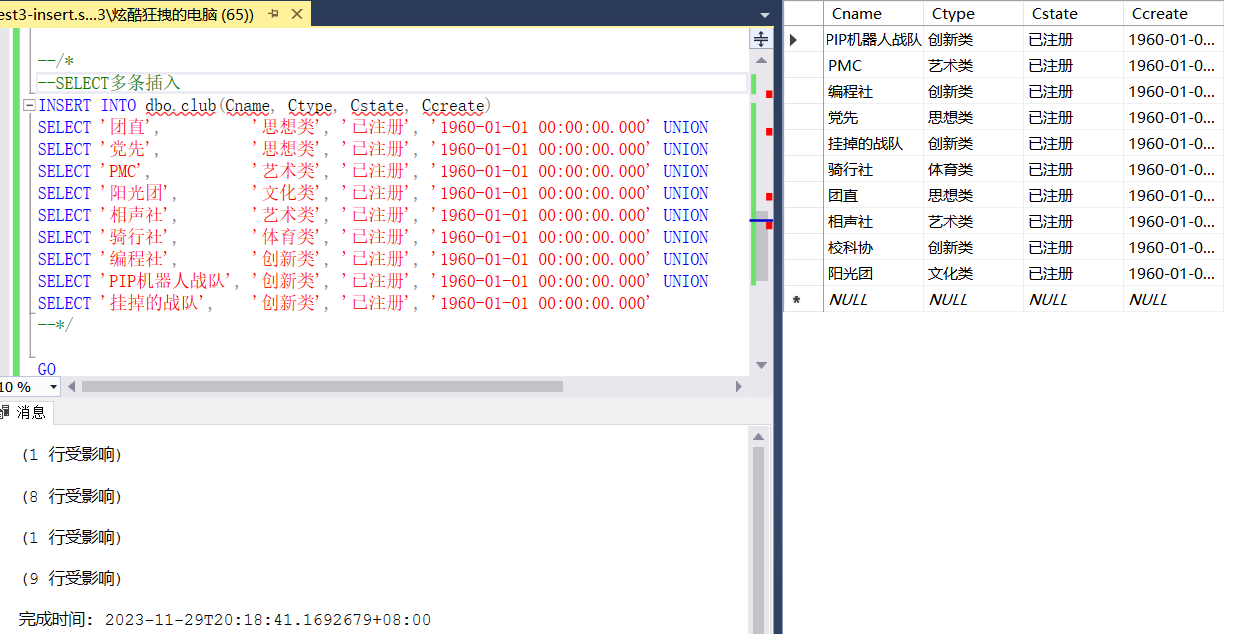


图3.4 SELECT多条插入

（2）在openGauss中插入数据

在openGauss中插入数据的脚本代码，和在SQL server中是一致的，详见下文图示代码和结果，不再单独列出，统一列出如图3.5-3.6所示

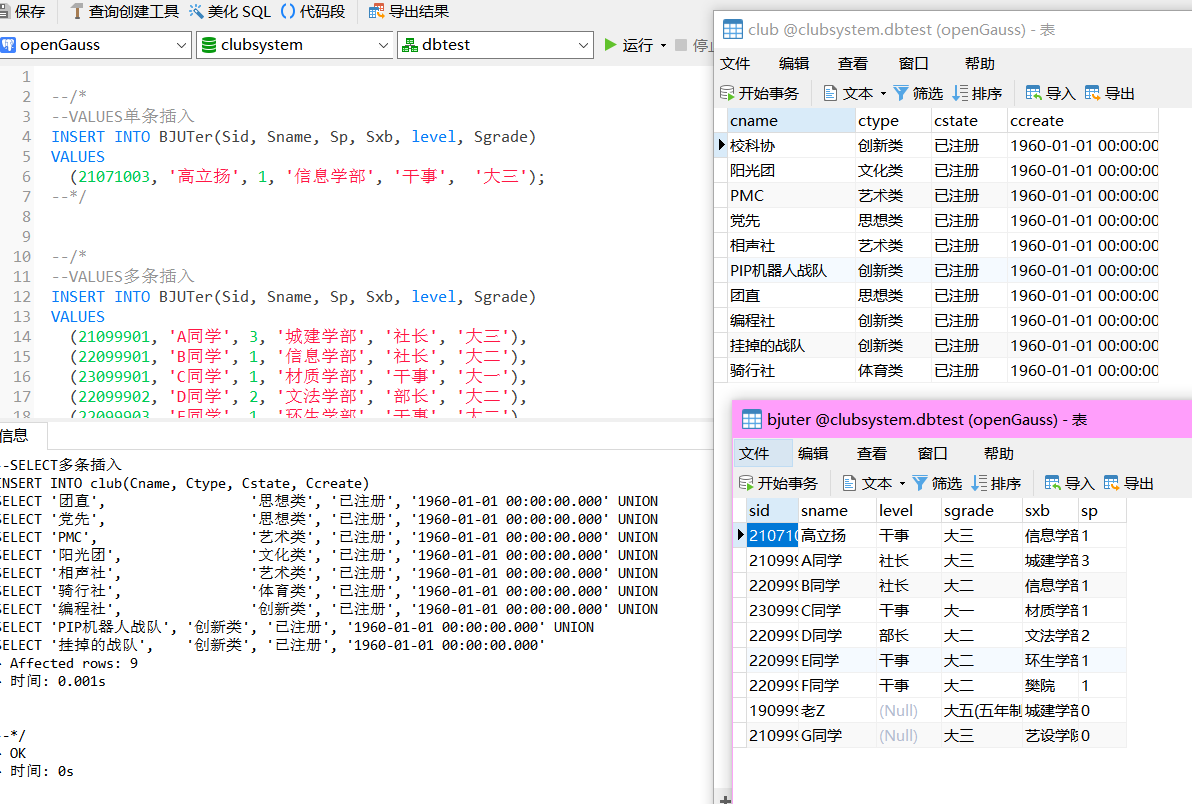


图3.5 VALUES单多 & 全部结果

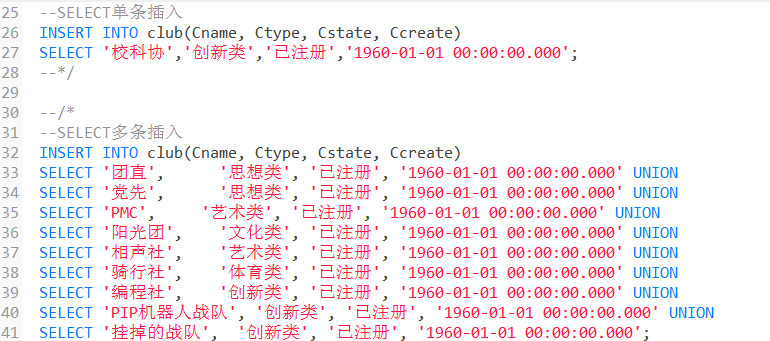


图3.6 SELECT单多

2.2 修改数据

（1）在SQL server中修改数据

在SQL server中，采用UPDATE语句进行数据的修改，也就是更新数据。UPDATE后面写表名，再后面写SET关键字，后面根据情况用WHERE或FROM等进行指定。在修改的时候，分单表修改和多表修改进行实验。

如图3.7所示，在2.1中，A同学所在学部为城建学部，而现在修改其为理学部

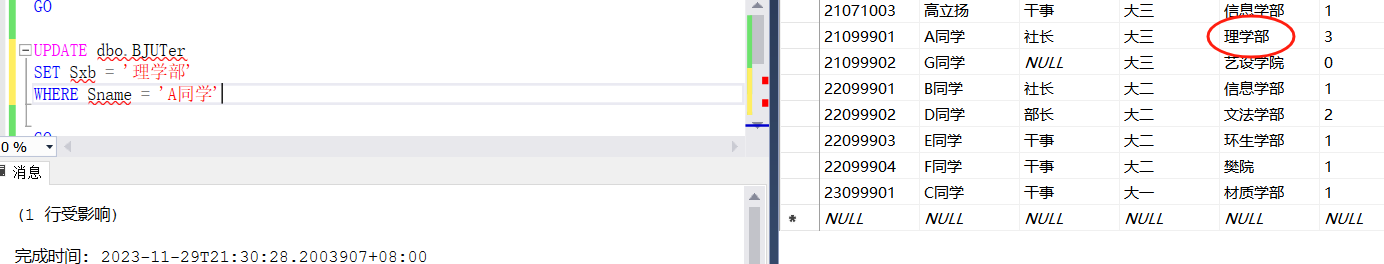


图3.7 A同学成功转学部

如图3.8，此图为修改前的activity表，如图3.9所示代码和修改结果，成功将表中Aplace为‘实训楼’和‘信息楼’的活动时间改为2023-11-30 19:07:59.000



图3.8 修改前

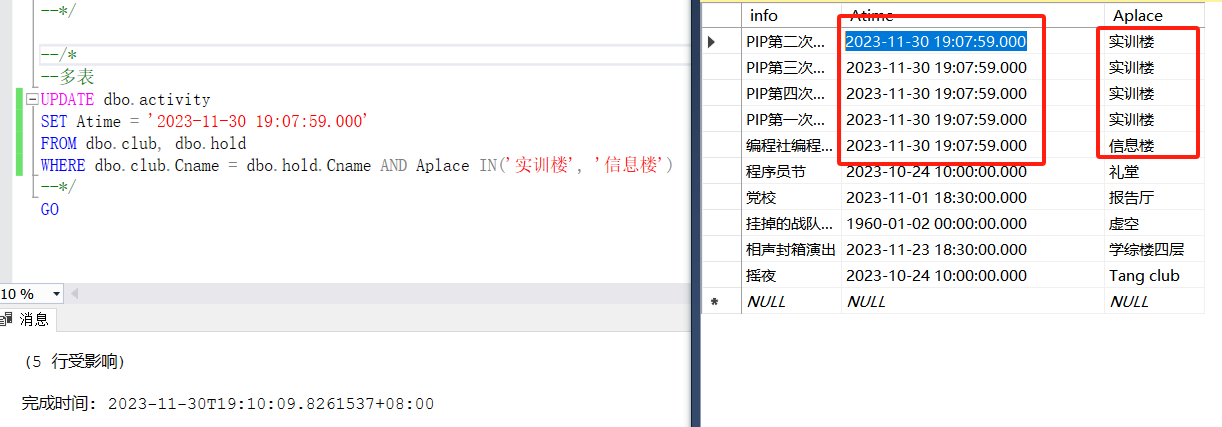


图3.9 修改后

（2）在openGauss中修改数据

在openGauss中，进行数据修改的操作同SQL server，详见下面的代码和结果图

如图3.10所示，UPDATE操作进行单表修改成功

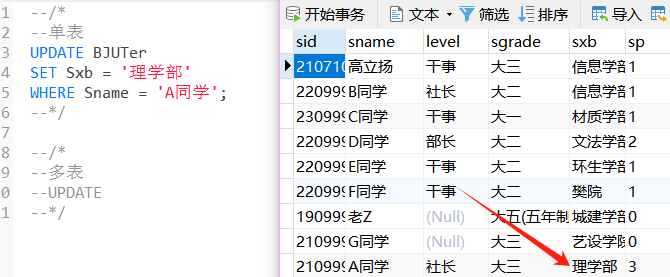


图3.10 修改成功

如图3.11所示，UPDATE操作进行多表修改成功

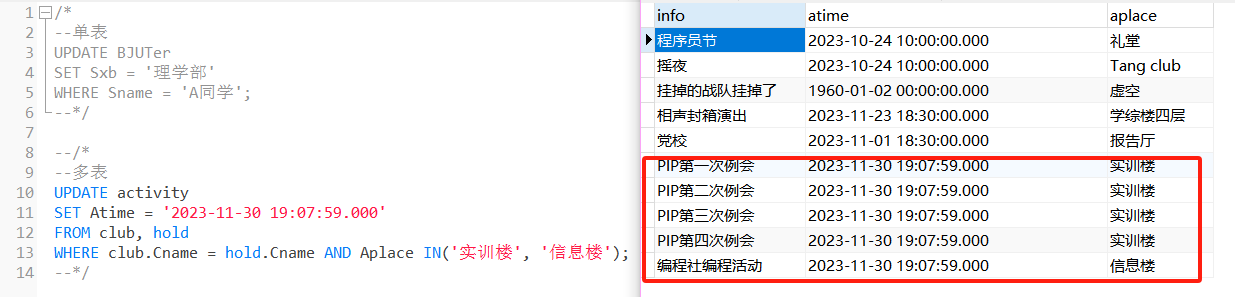


图3.11 修改成功

2.3 删除数据

（1）在SQL server中删除数据

删除数据可以用DELETE FROM ... WHERE来实现，根据实验要求，也需要考虑单表和多表的情况

如下图3.12所示，老Z同学存在于BJUTer表中，由于老Z同学面临毕业，因此不会再有参加社团的可能了，于是决定将其移出数据库，代码和结果如图3.13所示，老Z成功移出数据库，单表删除实验结束



图3.12 老Z还未移出数据库

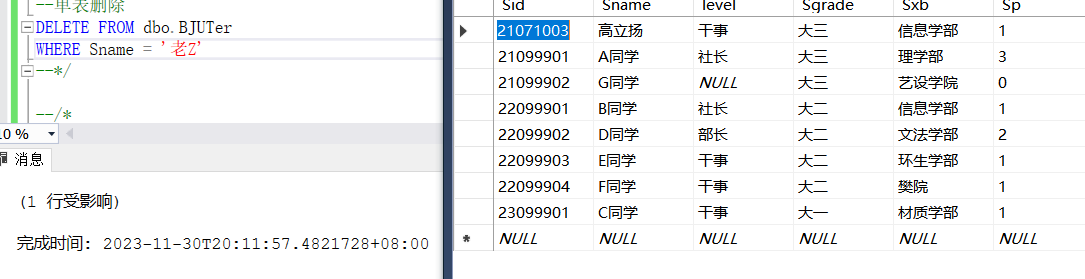


图3.13 老Z移出数据库

activity表中的全部活动及时间如下图3.14所示，下面要参照该表，对hold表中记录的社团举办的活动进行删除，删除2023年9月15号之前举办的活动，代码及结果如图3.15所示



图3.14 activity内容

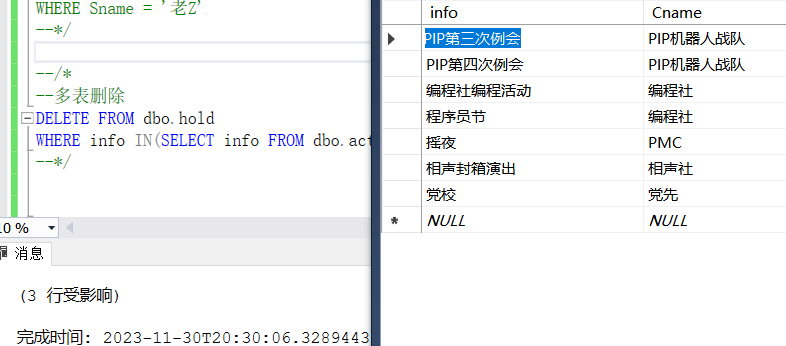


图3.15 删除了三个活动

（2）在openGauss中删除数据

在openGauss中删除数据的操作和SQL server中类似，详见下面代码和结果展示，如图3.16



图3.16 openGauss删除数据

##### 实验中出现的问题及分析

1. 假设向A表插入数据，而A表某个数据V引用了B表的K，那么V的数值不得超出B表K的范围，否则会报错，这就警示我们，先向被引用的表中添加数据，再向引用外键的表添加数据。这里并没有截图，是因为提前有同学询问我这个问题，而询问的时候我还没有做到这里，后来我们通过问其他同学，读学长的报告明白了这个问题的起因，因此在进行实验三的时候我避免了本问题，故没有截图

报错内容：INSERT语句与FOREIGN KEY约束“FK\_....”冲突。该冲突发生于数据库“XXX”,表XXX

1. openGauss中的LIKE '[0-9][0-9][0-9][0-9][0-9][0-9][0-9][0-9]'似乎不能像SQL server那样达到预期的功能，因此为了顺利完成实验，选择删掉此约束。不过在进行时间判断时，仍可以通过字符串比大小来完成，因此删除此约束并不影响openGauss中时间数据的正常操作
2. POSTGRESQL中一个中文字符占3字节，因此要调整VARCHAR的字节

##### 实验中未解决的问题

（1）openGauss如何书写约束，使得这个约束能达到SQL server中LIKE '[0-9][0-9][0-9][0-9][0-9][0-9][0-9][0-9]'的功能？

##### 思考题

Q1：OPENGAUSS和SQL Server 2012 提供了哪些类型的完整性约束？

A1：NOTNULL(非空)约束、CHECK(检查约束)、UNIQUE(唯一束)、PRIMARYKEY (主键约束)、FOREIGNKEY (外键约束)和DEFAULT(默认约束)

Q2：DELETE 语句与DROP TABLE语句有何不同？

A2：前者仅删除表中的元组，而表的模式依然存在；后者删除整个表，它将不复存在于数据库中

##### 心得体会

在本实验进行之初，我和许多同学进行了讨论和交流，了解到并避免了很多“坑”，也要感谢学长留下的报告，也能防止我“踩坑”。通过本实验，我学到了使用SQL语句进行数据的增删改，并在实践的过程中体会到数据库设计是多么的困难——设计人员要尽可能多地去设想约束，站在用户的角度考虑到多种意外情况！

#### 实验四 数据的检索 -- 第一部分

##### 实验目的

单表查询的实验是使用SELECT语句从单一基本表查询数据，主要目的是：

1）学会SELECT子句各种基本用法。

2）熟悉单表查询中各种WHERE条件的使用方法。

3）掌握常用的聚合函数的用法。

4）掌握分组统计的概念，熟悉GROUP BY 子句以及HAVING子句的基本用法。

5）掌握结果集输出时的各种排序方法，ORDER BY子句的常用方法。

##### 实验内容

2.1 SELECT查询

（1）在SQL server中SELECT查询

SQL server中的查询需要用到SELECT ... FROM ... WHERE，也就是从某表中选择某些属性，这些属性满足什么要求。如图4-1.1所示，查询“PIP机器人战队”的相关信息

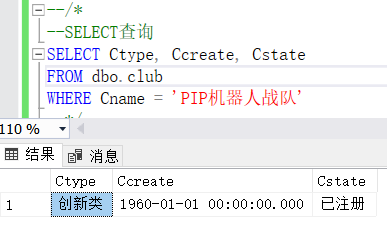


图4-1.1 SELECT查询

（2）在openGauss中SELECT查询

同SQL server，查询结果如图4-1.2所示

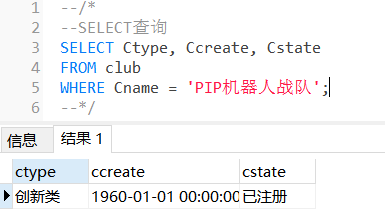


图4-1.2 SELECT查询

2.2 聚集函数查询

（1）在SQL server中聚集函数查询

聚集函数有很多种，比如SUM,AVG,COUNT等，下面对PIP战队举办的活动进行计数，如图4-1.3所示，由于在实验三中删除了hold表中PIP战队举办的两个活动，因此这里计数为2

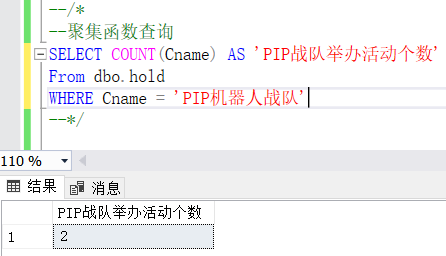


图4-1.3 PIP战队举办活动个数统计

（2）在openGauss中聚集函数查询

同上所述，进行查询结果如图4-1.4所示

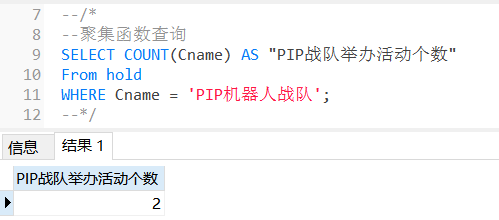


图4-1.4 PIP战队举办活动个数统计

2.3 DISTINCT去重

（1）在SQL server中DISTINCT去重

在SQL server中，对hold表针对Cname进行DISTINCT去重，目的是挑选出举办过活动的社团的名称，如图4-1.5所示

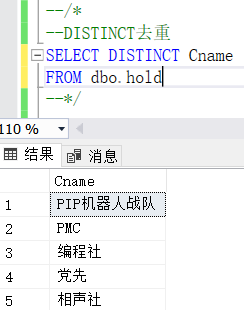


图4-1.5 社团去重查询

（2）在openGauss中DISTINCT去重

操作同SQL server，结果如图4-1.6所示

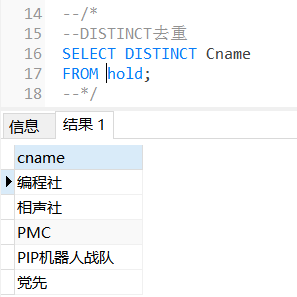


图4-1.6 社团去重查询

2.4 ORDER BY子句查询

（1）在SQL server中DISTINCT去重

ORDER BY可以将查询结果按照子句后面的条件，进行对应升降序的排序，下面针对BJUTer，按照Sid升序，按照Sp降序进行多列查询，如图4-1.7所示

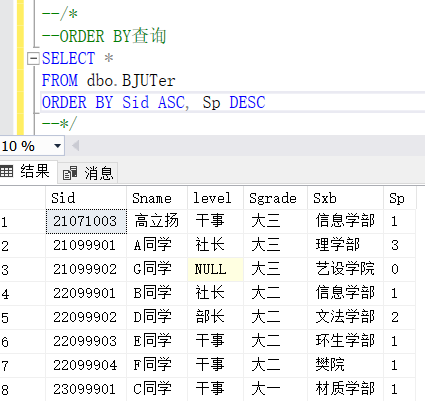


图4-1.7 BJUTer查询

（2）在openGauss中DISTINCT去重

操作同SQL server，结果如图4-1.8所示



图4-1.8 BJUTer查询

2.5 使用范围条件、算术运算符、逻辑运算符查询

（1）在SQL server中使用范围条件、算术运算符、逻辑运算符查询

对activity表进行如下查询：查询Atime在2023-09-07 00:00:00.000到2023-09-20 23:59:59.999之间（范围条件）（算术运算符），且（逻辑运算）Aplace是实训楼的活动信息，代码和运行结果如下图4-1.9所示

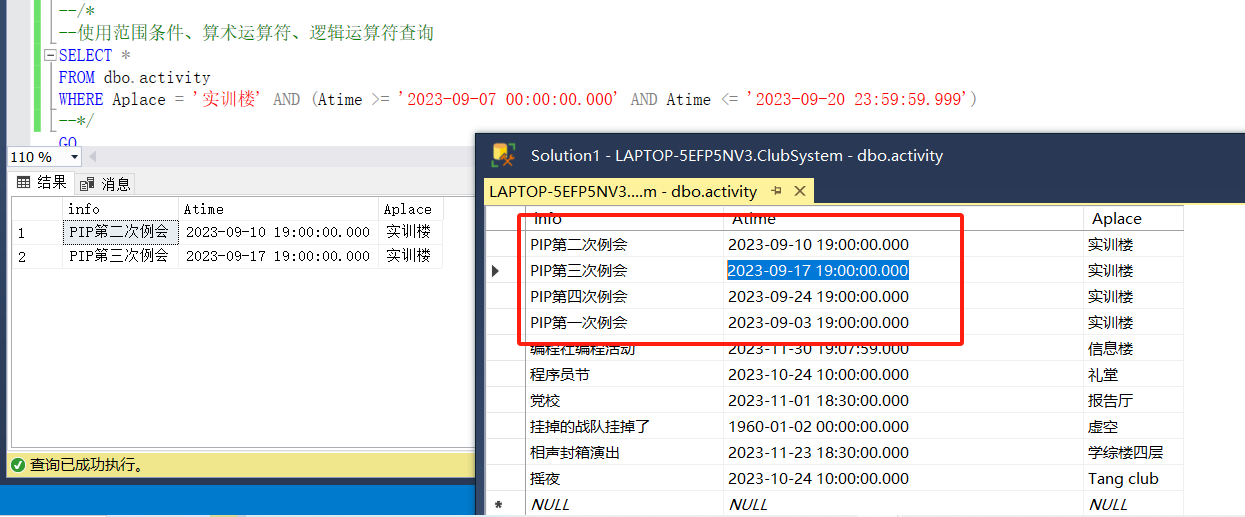


图4-1.9 使用范围条件、算术运算符、逻辑运算符查询

（2）在openGauss中使用范围条件、算术运算符、逻辑运算符查询

操作同SQL server，结果如图4-1.10所示



图4-1.10 使用范围条件、算术运算符、逻辑运算符查询

2.6 使用空值条件查询

（1）在SQL server中使用空值条件查询

对BJUTer表，查询没有参加过社团的同学，那么就需要查看level是否为NULL，代码和结果如下图4-1.11所示

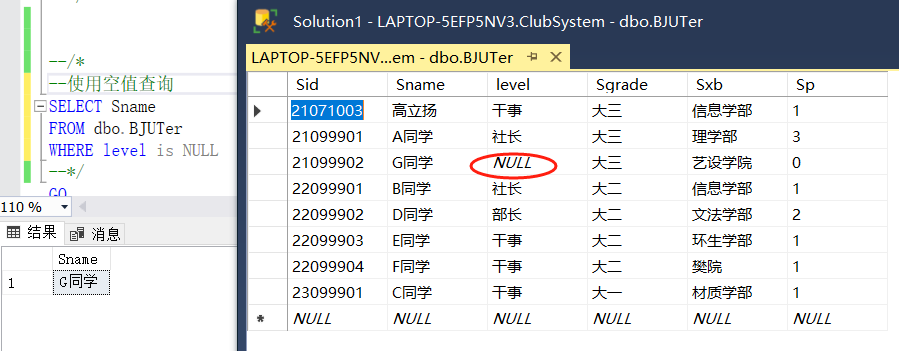


图4-1.11 使用空值条件查询

（2）在openGauss中使用空值条件查询

操作同SQL server，结果如图4-1.12所示



图4-1.12 使用空值条件查询

2.7 字符串的模式匹配查询

（1）在SQL server中字符串的模式匹配查询

用字符串匹配，查询activity表中开会性质的活动，代码和结果如图4-1.13

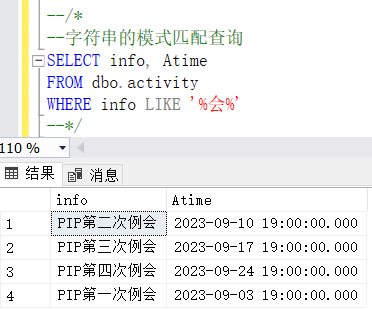


图4-1.13 字符串的模式匹配查询

（2）在openGauss中字符串的模式匹配查询

操作同SQL server，结果如图4-1.14所示



图4-1.14 字符串的模式匹配查询

2.8 GROUP BY & HAVING查询

（1）在SQL server中字符串的模式匹配查询

统计“创新类”社团举办活动的总数，代码和结果如图4-1.15所示，进一步筛选出举办活动总数>3的社团，代码和结果如图4-1.16所示，由于需求特殊，这里我其实用的是多表查询，与该实验“单表”的要求冲突，但是由于本小实验重点在于GROUP BY 和 HAVING的使用，因此保留子查询

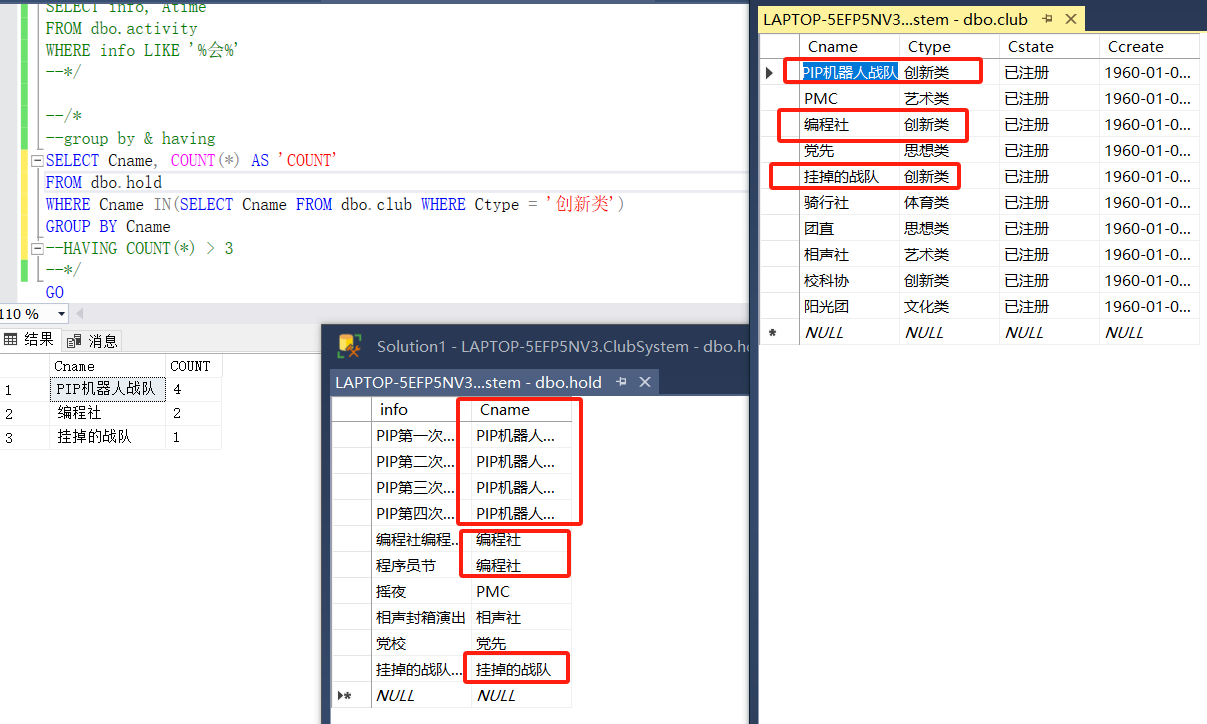


图4-1.15 仅用GROUP BY

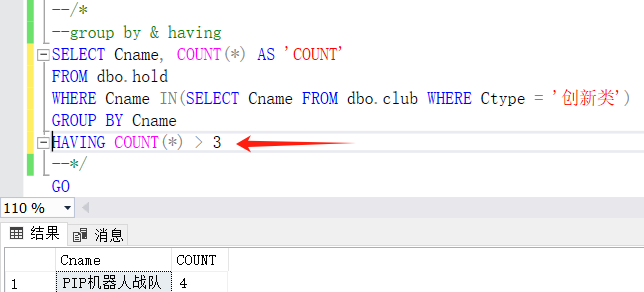


图4-1.16 使用HAVING

（2）在openGauss中字符串的模式匹配查询

操作同SQL server，需要注意AS后字符串用双引号而不是单引号，结果如图4-1.17-18所示

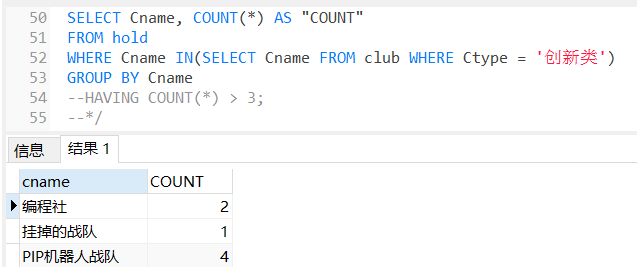


图4-1.17 仅用GROUP BY

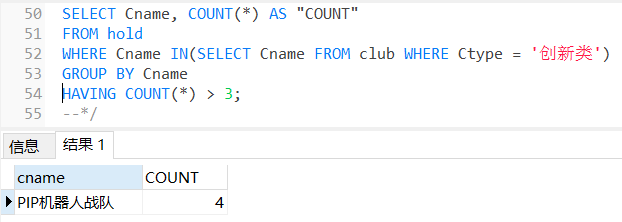


图4-1.18 使用HAVING

##### 实验中出现的问题及分析

1. openGauss中AS后面的字符串不可以用单引号括起来，而是需要用双引号括起来，否则会发生语法错误如图4-1.所示

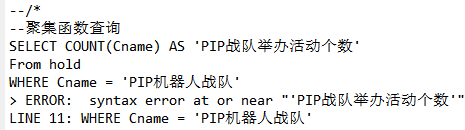


图4-1.19 错误提示

##### 实验中未解决的问题

本实验暂无未解决的问题

##### 思考题

Q1：什么是空值？

A1：数据库中，空值表示值未知。空值不同于空白或零值。没有两个相等的空值。空值一般表示数据未知、不适用或将在以后添加数据

Q2：为什么空值不用等号判定？

A2：比较两个空值或将空值与任何其他值相比均返回未知，这是因为每个空值均为未知。所以不用等号进行判定

Q3：聚合函数可以出现在什么子句中

A3：聚合函数可以出现在 SELECT 子句、HAVING 子句中，通常结合GROUP BY使用

Q4：什么情况使用HAVING？

A4：使用了 GROUP BY 子句后，如果还需要选择数据，就用 HAVING

##### 心得体会

通过本实验，我掌握了多种单表查询方法，还有排序与聚集的技巧，在学习过程中，我浏览过不少资料，来辅助完成实验。本实验是在数据库实验中，第一次需要根据需求来大量构思查询语句的实验，也是实验学时占比最大的实验，之前在课上学习到的知识，在这里得以验证和反思，加强了知识的记忆和理解，我受益良多！

#### 实验四 数据的检索 -- 第二部分

##### 实验目的

多表查询的实验是使用查询语句从多个基本表或视图查询数据，包含连接查询（内连接）、集合查询以及子查询3种查询方法，本实验主要目的是：

1）学会内连接查询的表示方法（标准表示法或简约表示法均可），以及自连接的表示法。

2）学会集合查询的达，包括UNION、INTERSECT和EXCEPT的表达，集合运算的“并兼容”问题。

3）学会子查询即嵌套查询的使用方法，包括3种形式引入子查询的方法：[NOT] IN、 比较运算符与ALL|ANY 和EXISTS；理解相关子查询和独立子查询的概念，学会相关子查询的表达方法。

4）学会上述3种多表查询方法的综合应用。

5）学会上述3种多表查询与GROUP BY 子句以及ORDER BY 子句的联合使用。

6）深入理解主键、外键的概念。

7）深入理解实体完整性约束与参照完整性约束的概念。

学习使用SELECT语句在多张基本表中查询各类信息。熟悉WHERE条件的表达、DISTINCT的使用、连接条件与选择条件的表达。理解连接运算。

##### 实验内容

2.1 内连接查询

（1）在SQL server中使用内连接查询

使用双重内连接，查询举办时间在2023-10-01 00:00:00.000及以后的活动及举办社团的信息，代码和结果如图4-2.1所示

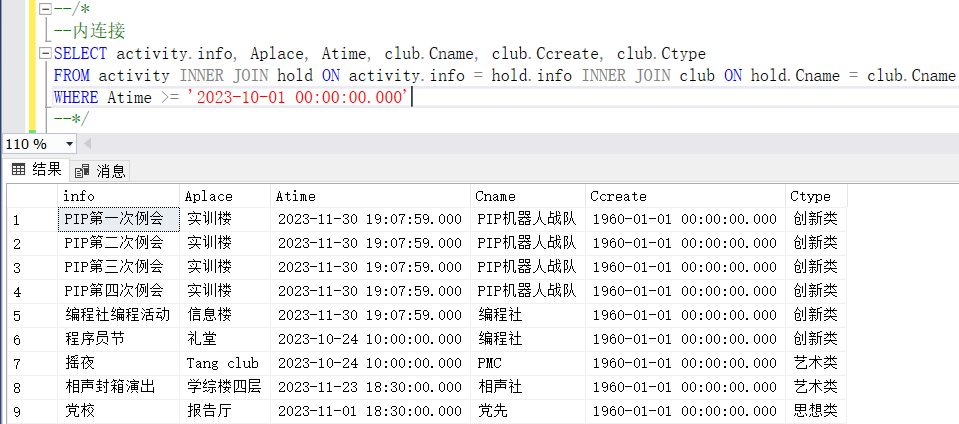


图4-2.1 内连接查询

（2）在openGauss中使用内连接查询

操作同SQL server，结果如图4-2.2所示

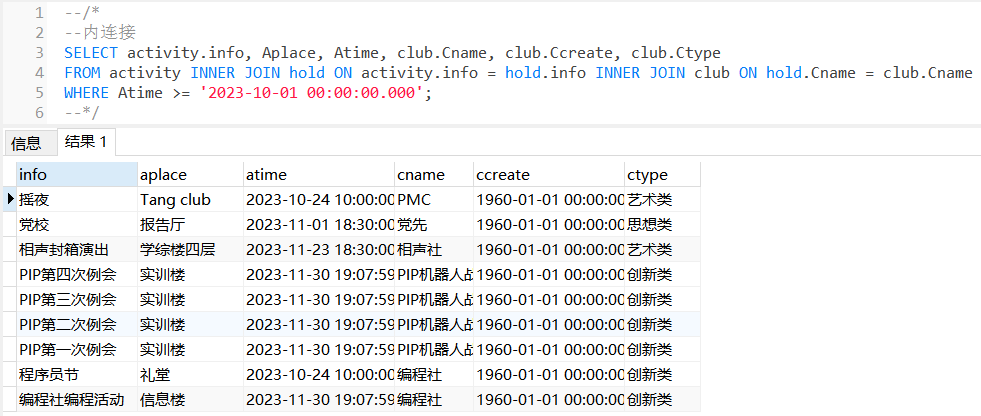


图4-2.2 内连接查询

2.2 派生表查询

（1）在SQL server中使用派生表查询

SELECT语句返回的表就是派生表。使用派生表及重命名，来查询实验四-第一部分中2.8功能相同的查询，代码和结果如图4-2.3所示

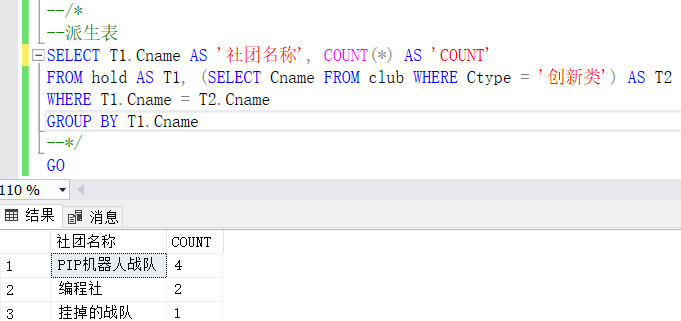


图4-2.3 派生表查询

（2）在openGauss中使用派生表查询

操作同SQL server，结果如图4-2.4所示

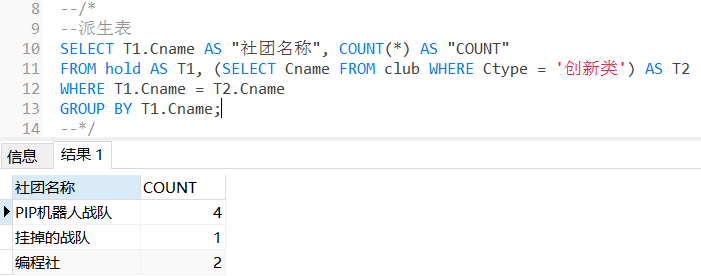


图4-2.4 派生表查询

2.3 自连接查询

（1）在SQL server中使用自连接查询

使用自连接的方法，查询BJUTer表中，职位相同的同学，代码和结果如图4-2.5所示，用Sid对比来防止重复，简化结果

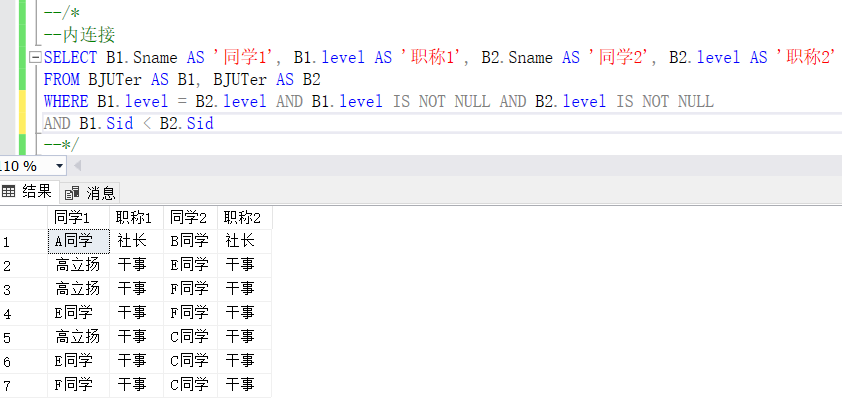


图4-2.5 自连接查询

（2）在openGauss中使用自连接查询

操作同SQL server，结果如图4-2.6所示



图4-2.6 自连接查询

2.4 INTERSECT查询

（1）在SQL server中使用INTERSECT查询

顾名思义，INTERSECT求的是两个派生表的交集，可以寻找两个表相同的列。欲查询同在“PIP机器人战队”的同学，进行交集查询，代码和结果如图4-2.7

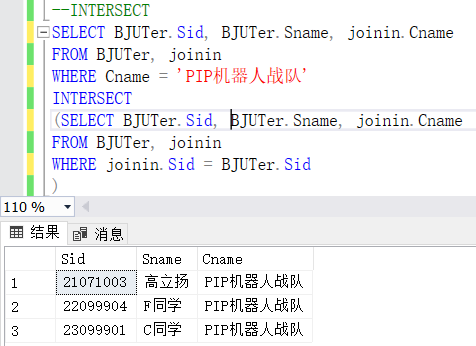


图4-2.7 INTERSECT查询

（2）在openGauss中使用INTERSECT查询

操作同SQL server，结果如图4-2.8所示

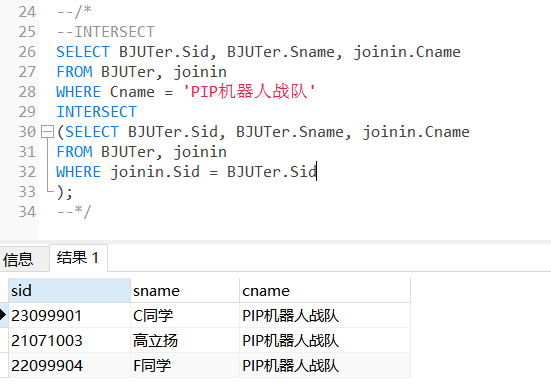


图4-2.8 INTERSECT查询

2.5 UNION查询

（1）在SQL server中使用UNION查询

UNION是合集，下面查询BJUTer中，高立扬同学和A同学都参加了哪些社团，代码和结果如图4-2.9所示

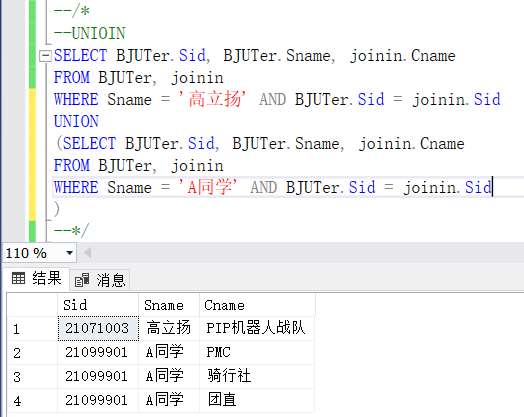


图4-2.9 UNION查询

（2）在openGauss中使用UNION查询

操作同SQL server，结果如图4-2.10所示

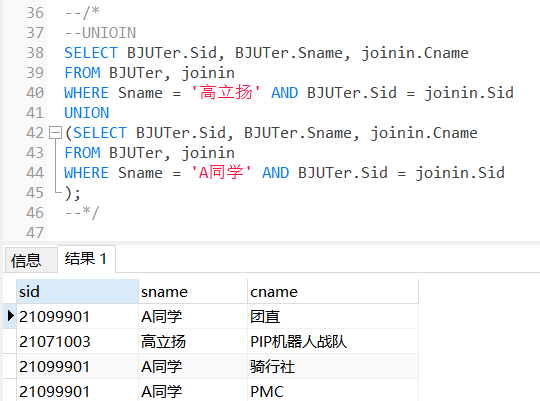


图4-2.10 UNION查询

2.6 EXCEPT查询

（1）在SQL server中使用EXCEPT查询

EXCEPT是差集。欲查询数据库中全部BJUTer没参加的战队，代码和结果如图4-2.11所示

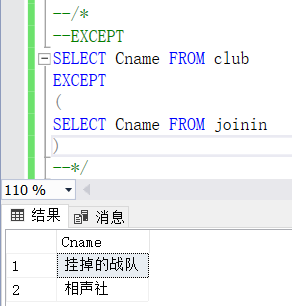


图4-2.11 EXCEPT查询

（2）在openGauss中使用EXCEPT查询

操作同SQL server，结果如图4-2.12所示

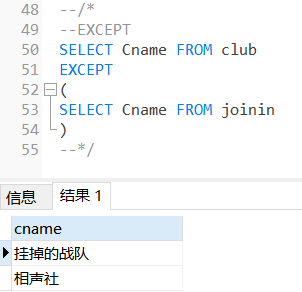


图4-2.12 EXCEPT查询

2.7 IN子查询

（1）在SQL server中使用IN子查询

利用IN子查询，寻找举办过活动的创新类社团，代码和结果如图4-2.13

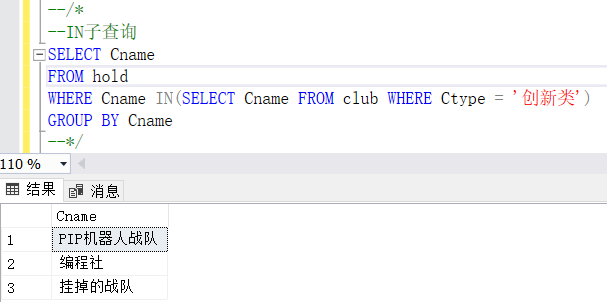


图4-2.13 IN子查询

（2）在openGauss中使用IN子查询

操作同SQL server，结果如图4-2.14所示

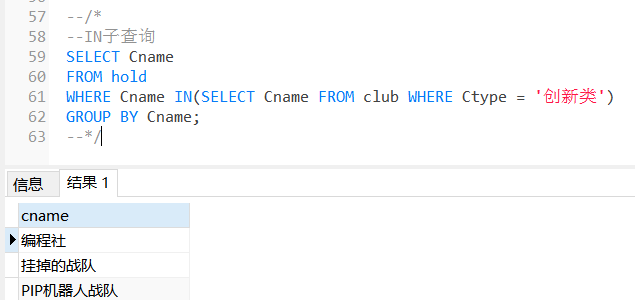


图4-2.14 IN子查询

2.8 ORDER BY 子句与连接查询

（1）在SQL server中使用ORDER BY 子句与连接查询

用ORDER BY子句，对2.1内连接查询进行扩展，代码和结果如图4-2.15

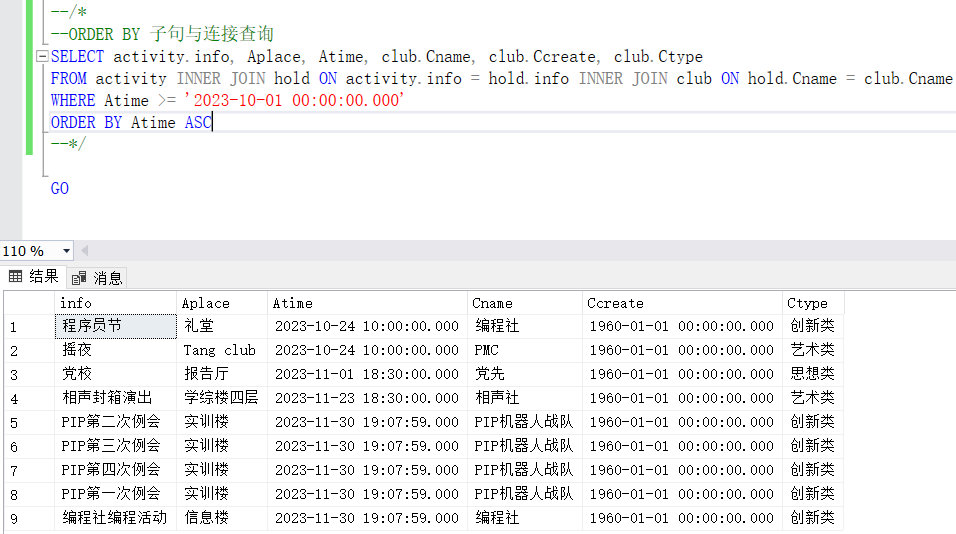


图4-2.15 ORDER BY 子句与连接查询

（2）在openGauss中使用ORDER BY 子句与连接查询

操作同SQL server，结果如图4-2.16所示

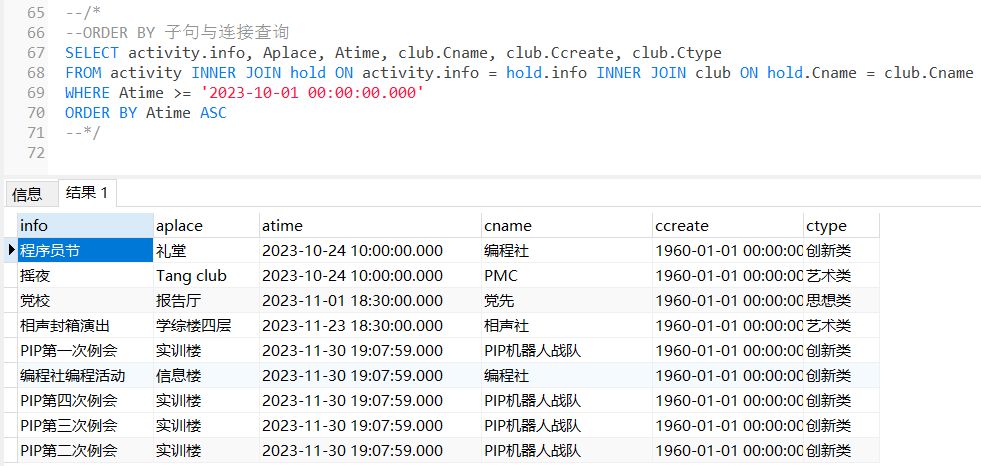


图4-2.16 ORDER BY 子句与连接查询

2.9 EXISTS查询

（1）在SQL server中使用EXISTS查询

EXISTS会检测是否存在满足条件的数据，下面查询是否存在满足举办活动数目大于3的社团，代码和结果如图4-2.17所示

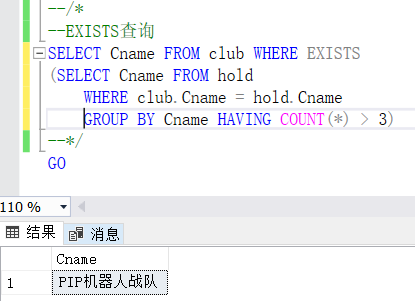


图4-2.17 EXISTS查询

（2）在openGauss中使用EXISTS查询

操作同SQL server，结果如图4-2.18所示

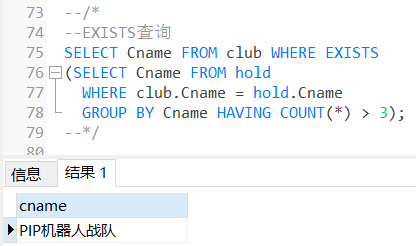


图4-2.18 EXISTS查询

2.10 ANY查询

（1）在SQL server中使用ANY查询

借助ANY，来查询指导社团数目比A同学参加社团数目多的teacher，经测试，此查询用ANY和ALL效果一样，代码和结果如图4-2.19所示

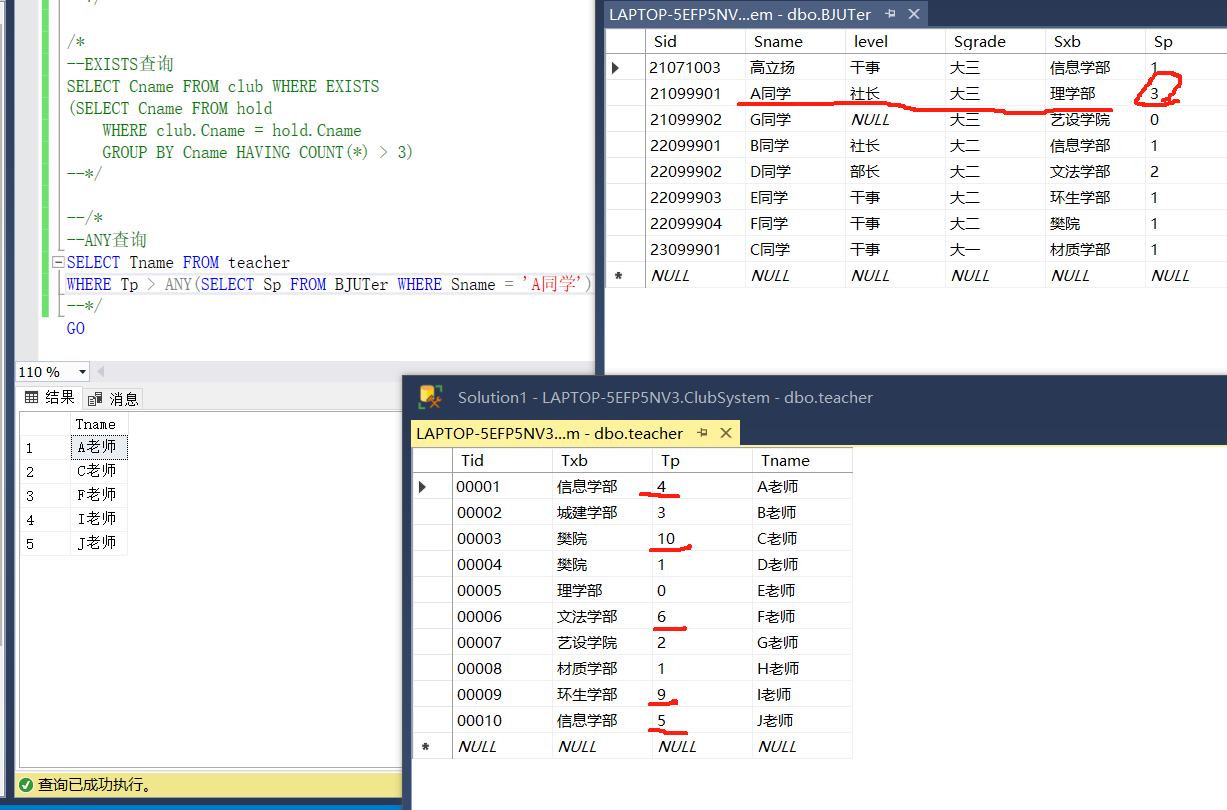


图4-2.19 ANY查询

（2）在openGauss中使用ANY查询

操作同SQL server，结果如图4-2.20所示

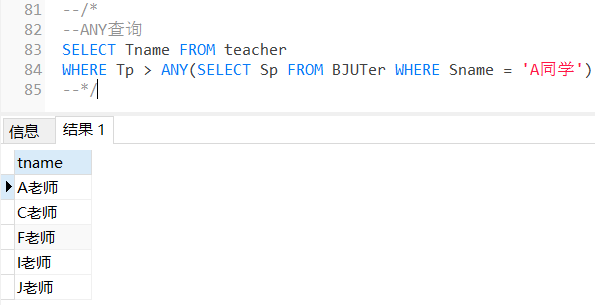


图4-2.20 ANY查询

2.11 综合查询应用

（1）在SQL server中综合查询应用

下面将用到EXISTS，对实验四-第一部分的最后一个实验进行拓展，以展现本实验四-第二部分学习成果。下面，对属于创新类且开设活动大于3次的社团，展示它开设过的活动，代码和结果如图4-2.21所示

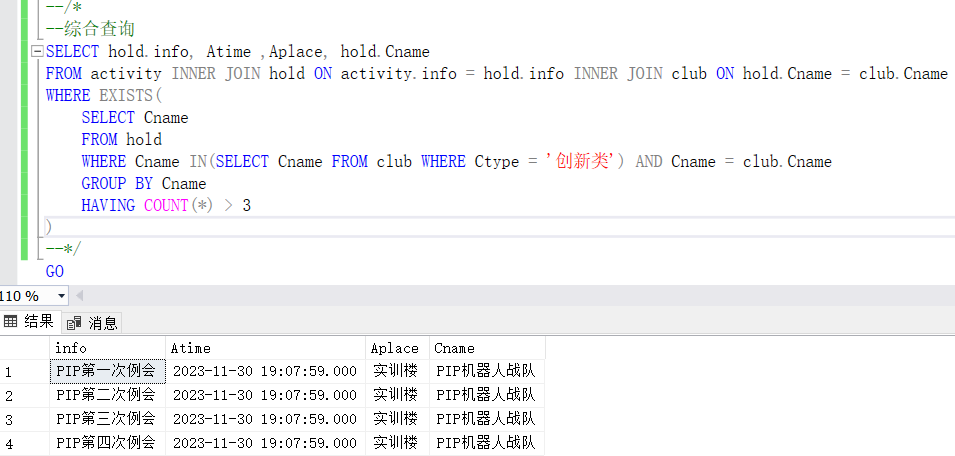


图4-2.21 综合查询应用

（2）在openGauss中综合查询应用

操作同SQL server，结果如图4-2.22所示

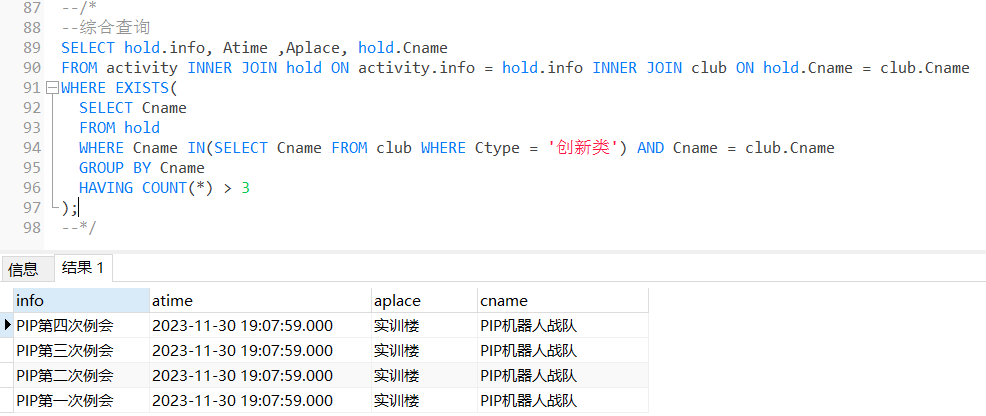


图4-2.22 综合查询应用

##### 实验中出现的问题及分析

本次实验未遇到问题

##### 实验中未解决的问题

本实验暂无未解决的问题

##### 思考题

Q1：连接条件一定是对应属性相等吗？

A1：不一定，还可以比大小

Q2：所有的查询都可以使用多表连接和子查询两种方法吗？

A2：结合我自己的认知和设想范围内的实践，我认为可以。情况不同，二种方法的适用性也不同，虽然二者可以互相实现，但是针对不同的应用问题，其效率也不同，这是需要具体问题具体分析的

##### 心得体会

通过本实验，我掌握到了子句查询，复杂的查询。随着实验的推进，我逐渐可以自行根据需求构思代码，遇到代码书写错误或者结果不在预期的情况，也可以自行修改并实现预期，我对课堂知识的理解也再一次加深

#### 实验五 创建和删除视图

##### 实验目的

本实验主要是通过学习视图的相关知识，了解数据库对象——视图的作用，创建、修改、删除视图及视图加密等相关技术。具体要求如下：

1)掌握视图的基本概念，了解视图在数据库系统中的作用及原理。

2)掌握使用-SL进行视图的创建、修改和删除操作。

3)了解基于视图进行表数据的修改及其注意事项。

4)了解视图加密的方法。

##### 实验内容

2.1 创建视图

（1）在SQL server中创建视图

通过hold来连接activity和club，并依此创建视图，代码如图5.1所示

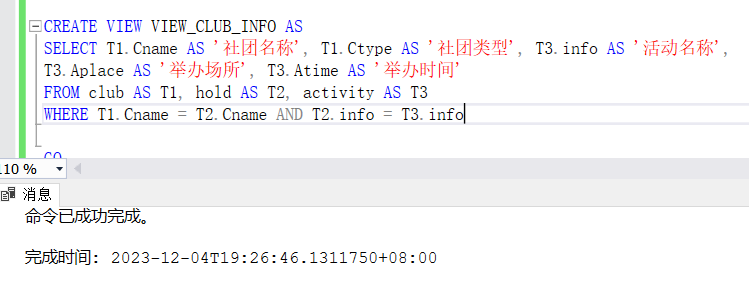


图5.1 创建视图

视图内容如图5.2，还可以通过SELECT \* FROM VIEW\_CLUB\_INFO来查看

图5.2 视图

（2）在openGauss中创建视图

操作同SQL server，结果如图5.3-4所示

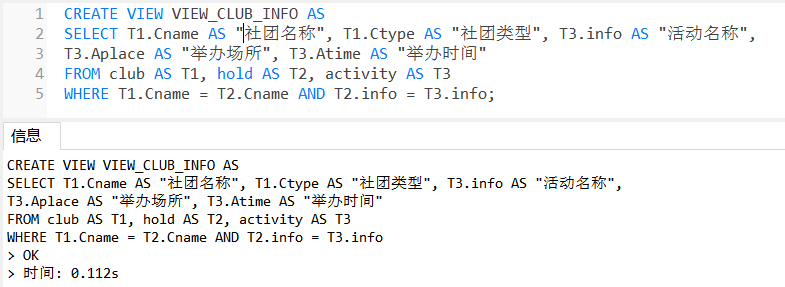


图5.3 创建视图



图5.4 视图

2.2 修改视图

（1）在SQL server中修改视图

可以通过ALTER VIEW来修改视图，下面展示为视图增加列的代码和结果如图5.5-6所示，减少列同理，不再赘述

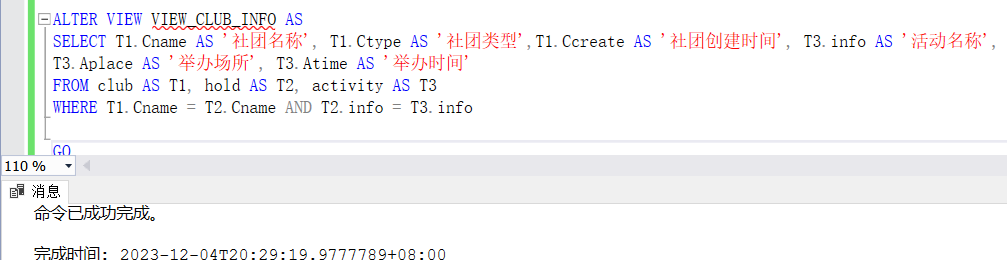


图5.5 修改视图



图5.6 修改后的视图

（2）在openGauss中修改视图

openGauss无法修改视图，只能先删除原有视图，再重新创建视图，在创建过程时做出改变。还未展示删除视图，本环节操作详见下文2.3 （2）

2.3 删除视图

（1）在SQL server中删除视图

代码和结果见图5.7-8

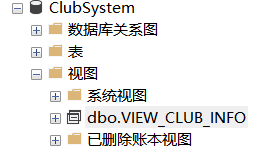


图5.7 删除前

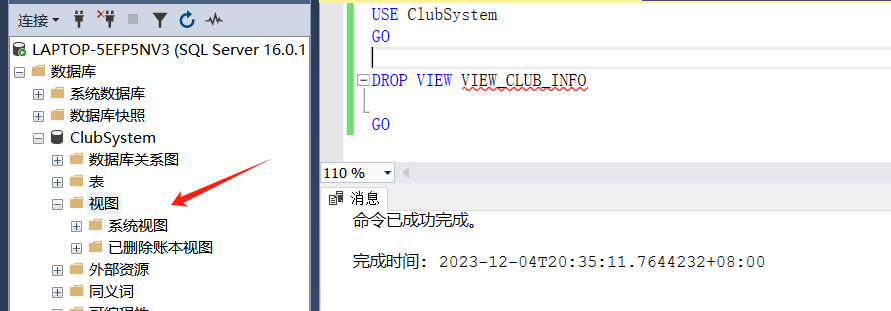


图5.8 删除后

（2）在openGauss中删除视图

操作同SQL server，并且删除后，重新创建视图，来模拟SQL server中的修改视图操作，结果如图5.9-11所示

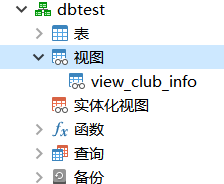


图5.9 删除前

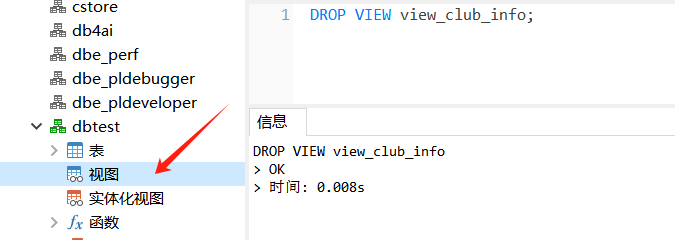


图5.10 删除后

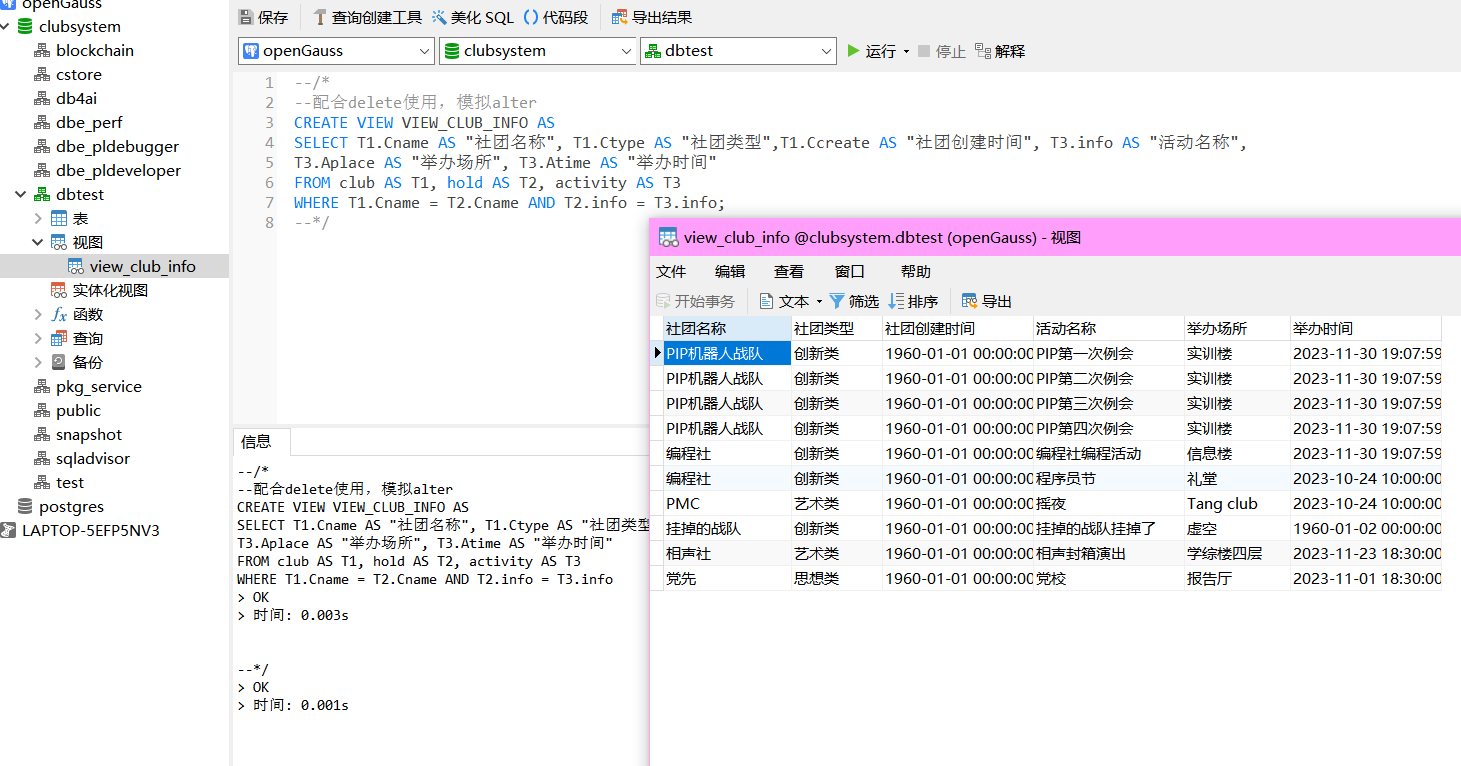


图5.11 delete+create模拟alter

2.4 T-SQL操作视图

（1）在SQL server中操作视图

操作视图分为：插入一条记录、删除一条记录和修改一条记录的内容，代码和结果如图5.12-14所示，可见，对视图的修改实际上是对基表的修改，因为视图仅能查看

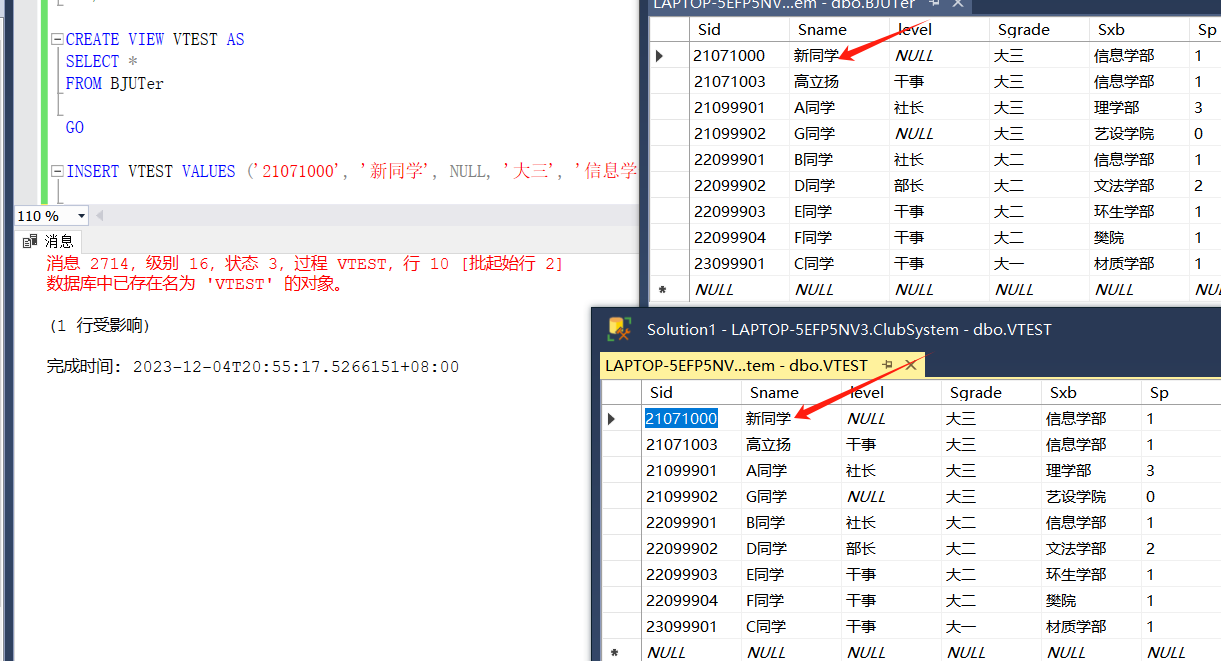


图5.12 插入一条记录

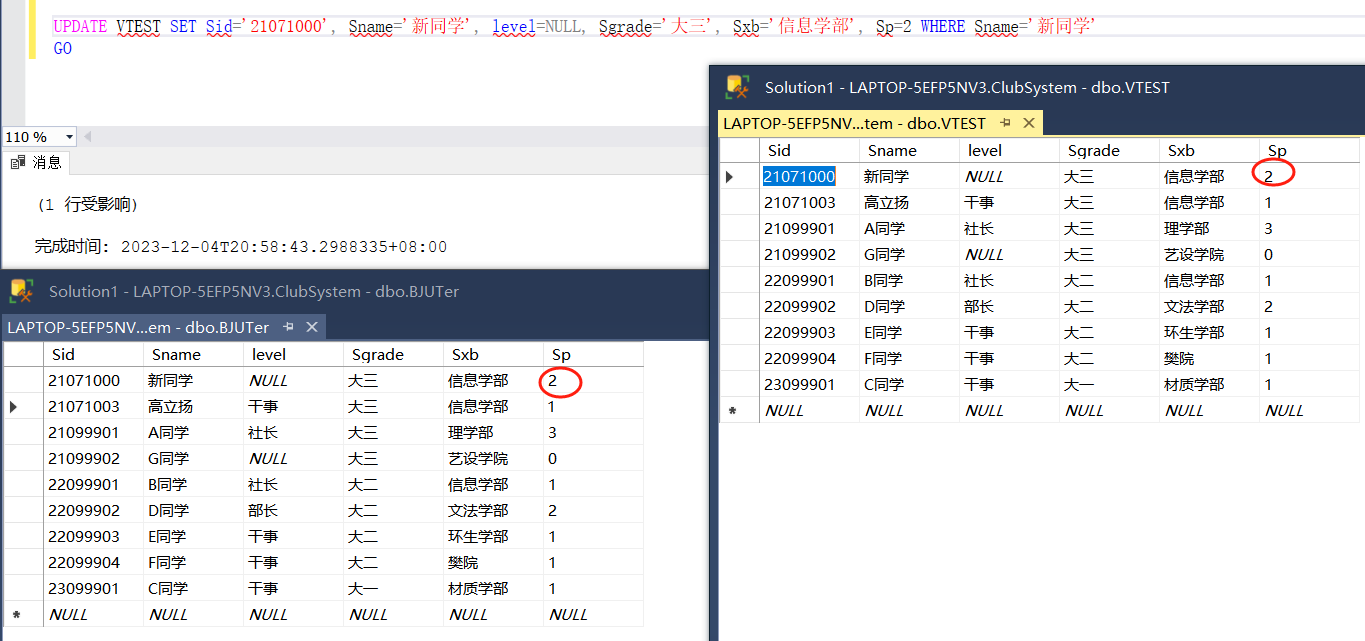


图5.13 修改一条记录

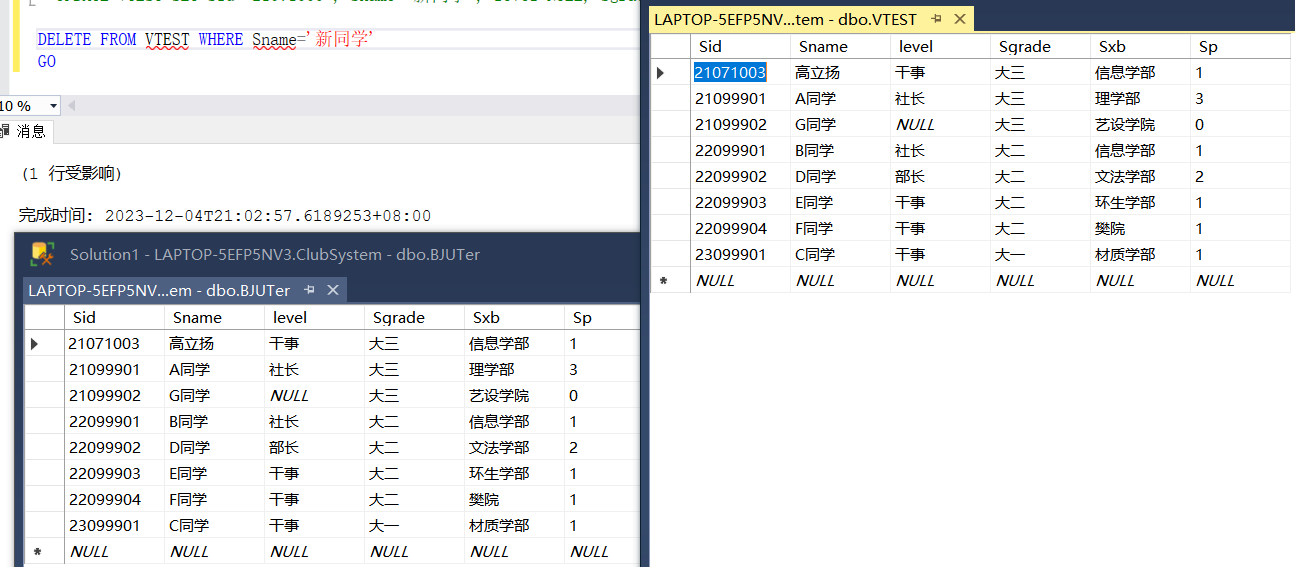


图5.14 删除一条记录

（2）在openGauss中操作视图

出于安全性考虑，openGauss不允许视图修改，报错如图5.15所示

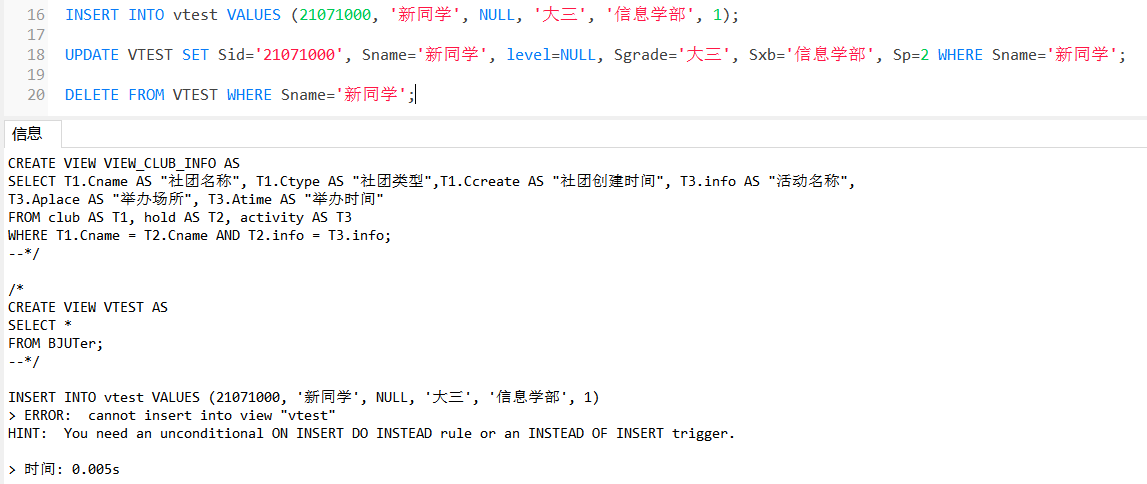


图5.15 报错

2.5 限制引用表的删除

（1）在SQL server中限制引用表的删除

在创建视图时，添加WITH SCHEMABINDING即可实现限制引用表的删除，如图5.16，进行上述视图创建，随后在视图删除引用表的时候发生错误

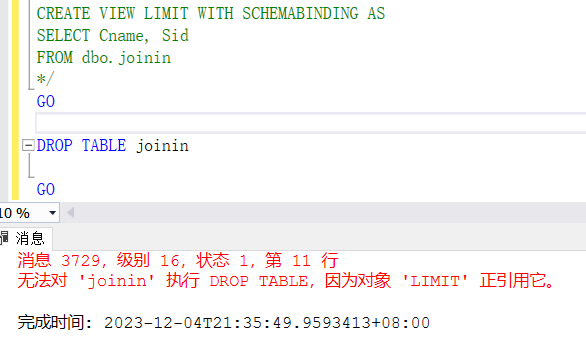


图5.16 无法删除

（2）在openGauss中限制引用表的删除

openGauss中没有SQL server中的WITH SCHEMABINDING，因为如图5.17可见，openGauss在创建视图时已经考虑到了安全性，用户无法通过常规途径直接删除关联基表

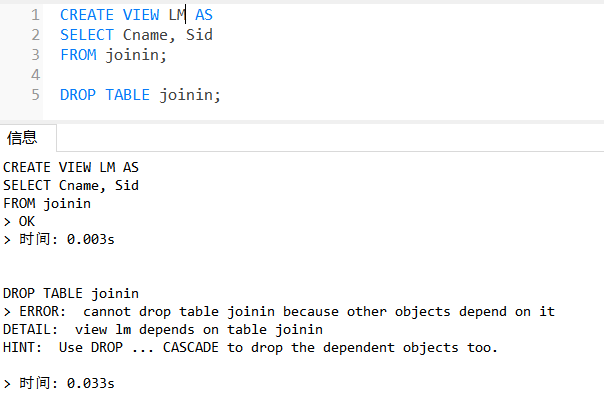


图5.17 无法删除

2.6 设置视图加密

（1）在SQL server中设置视图加密

如图5.18所示，创建一个加密视图，通过SELECT发现加密的视图无法查看其定义，说明该视图已经被加密

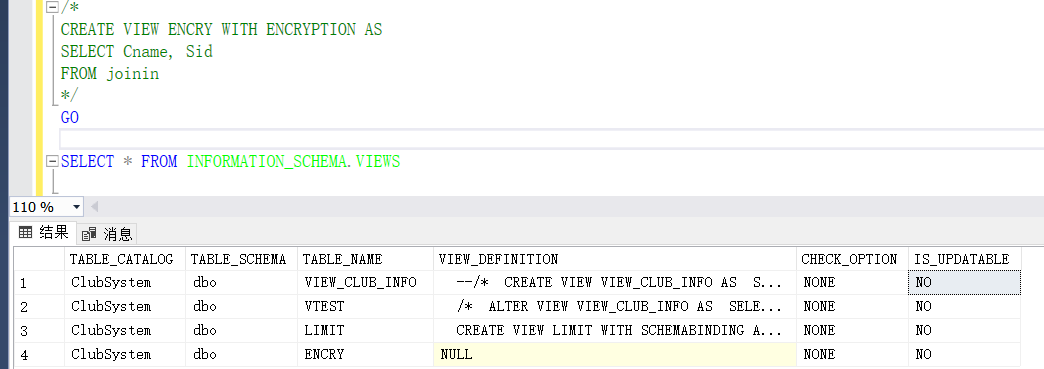


图5.18 已加密

（2）在openGauss中设置视图加密

openGauss中同样无法加密表，这也是因为其安全性，权限不足的用户无法查看对应的视图，如图5.19所示，dbtest用户有权查看自己的视图定义，无权查看系统视图定义，而在openGauss虚拟机中以omm用户身份进行查询时，可以查看系统视图的定义，如图5.20所示

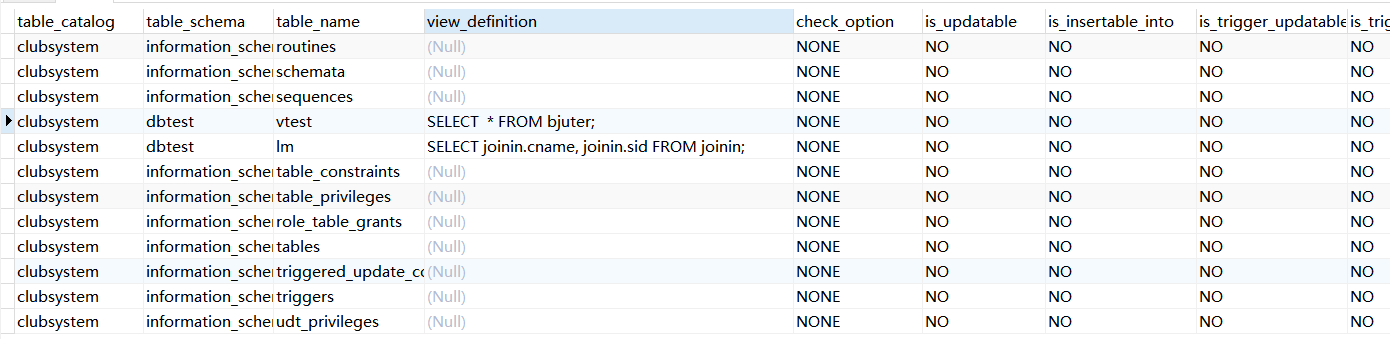


图5.19 权限不足

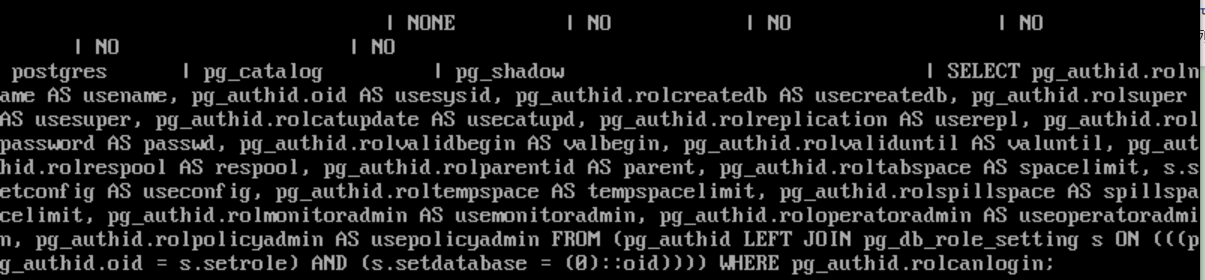


图5.20 权限充足

##### 实验中出现的问题及分析

1. 视图或函数 'VIEW\_CLUB\_INFO' 不可更新，因为修改会影响多个基表

如图5.21所示，无法对2.1创建的表进行插入记录，弹出错误，通过Stack Overflow中的一个帖子，我了解到了Any modifications, including UPDATE , INSERT , and DELETE statements, must reference columns from only one base table.也就是说，只有基表才能进行修改，而2.1创建的视图关联了三个表，因此无法修改

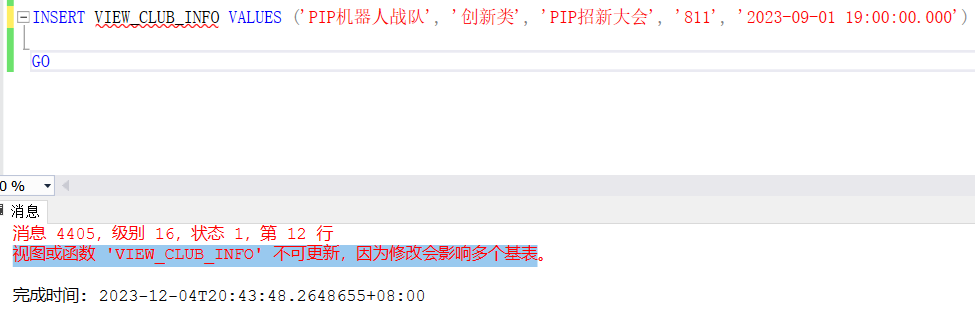


图5.21 错误

1. 无法将 视图'LIMIT' 绑定到架构，因为名称 'joinin' 对于架构绑定无效。名称必须由两部分构成，并且对象不能引用自身。

如图5.22所示，网上查阅资料发现需要写成dbo.joinin，而不是joinin

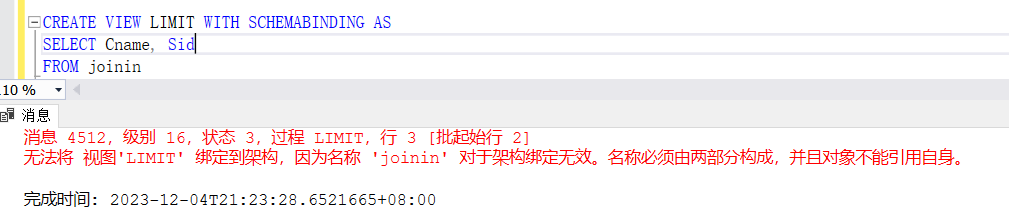


图5.22 错误

##### 实验中未解决的问题

本实验暂无未解决的问题

##### 思考题

Q1：视图与基本表有什么不同？

A1：

视图是一种虚拟表，可以包含来自多个表的数据，用于提供查看数据的特定视角；而基本表则是实际存储在数据库中的数据，由行（记录）和列（字段）组成

①视图是已经编译好的SQL语句，而基本表不是；

②视图没有实际的物理记录，而基本表有；

③视图是内容的一种窗口展示方式，只占用逻辑空间，不占用物理空间，而基本表则占用物理空间；

④基本表可以随时修改，但视图只能用创建的语句来修改；

⑤基本表属于全局模式中的表，是实表；视图属于局部模式的表，是虚表；

⑥视图的建立和删除只影响视图本身，不影响对应的基本表。

##### 心得体会

通过本实验，我学习到了什么是视图，通过两个数据库的对比操作，我认识并切身体会到了openGauss的“高安全”。视图作为引用表的存在，常用于展示功能，而不可以让低权限用户通过修改视图来破坏基表，从而毁坏数据库。因此需要对视图进行限制、加密等操作，这是保证数据库安全的必需

#### 实验六 创建和删除索引

##### 实验目的

本实验主要目的在于通过学习数据库索引的相关知识，了解数据库索引的结构、类型，创建方法以及索引的基本维护方法（重新生成索引和重新组织索引）。具体要求如下：

1）掌握数据库索引基本概念，以及索引的基本类型。

2）学会使用SQL创建、查看和修改索引。

3）学会使用SQL重新生成索引。

4）学会使用SQL重新组织索引。

##### 实验内容

2.1 设置UNIQUE约束

（1）在SQL server中设置视图加密

设置teacher表中的Tname为UNIQUE，可见系统自动生成了索引，类型是唯一、非聚集，如图6.1所示

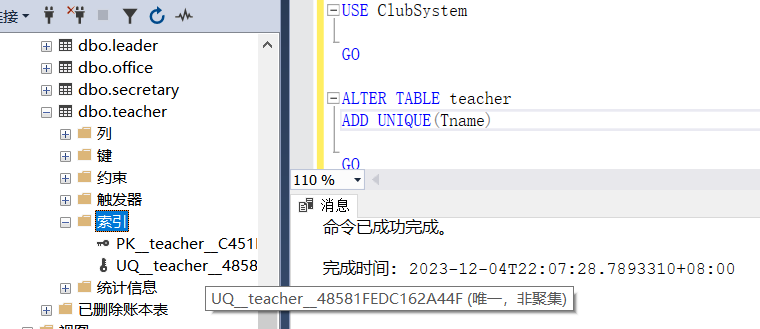


图6.1 索引

（2）在openGauss中设置视图加密

openGauss在添加UNIQUE时，系统不会自动创建索引，如图6.2所示，不生成索引的原因是ALTER TABLE / ADD UNIQUE will create implicit index "teacher\_tname\_key" for table "teacher"

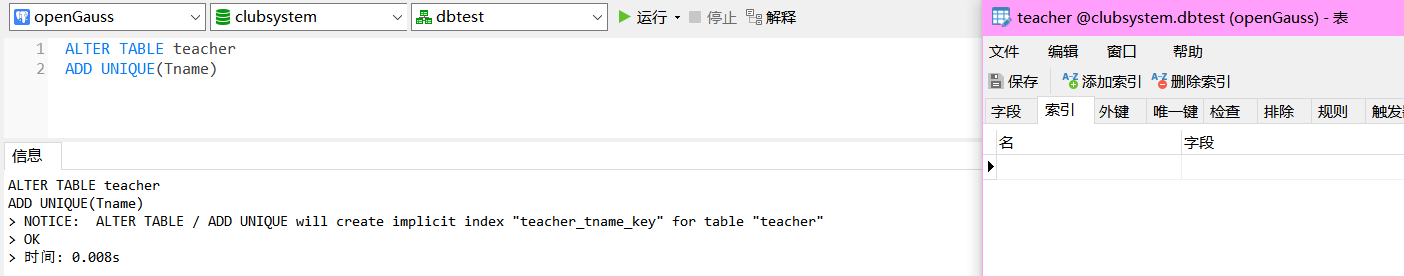


图6.2 无索引

2.2 创建聚集索引

（1）在SQL server中创建聚集索引

由于还没有学习删除索引，因此在SSMS中手动删除leader表的聚集索引，然后按照图6.3的操作重新创建，结果如图

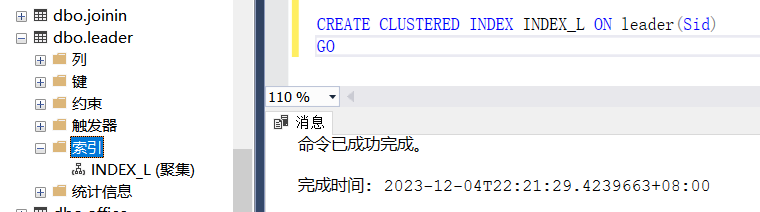


图6.3创建成功

（2）在openGauss中创建聚集索引

openGauss中没有CLUSTERED关键字，通过如图6.4所示的代码创建索引

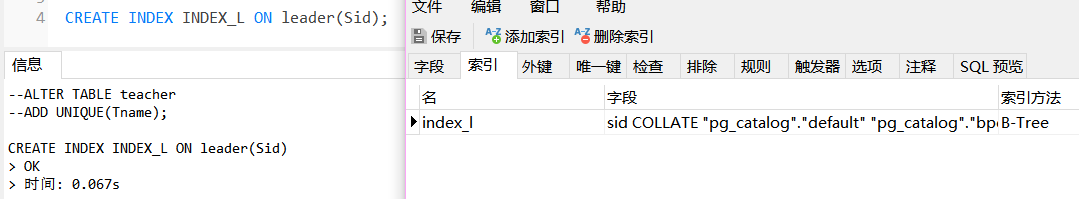


图6.4 创建索引

2.3 查看索引

（1）在SQL server中创建查看索引

如图6.5所示

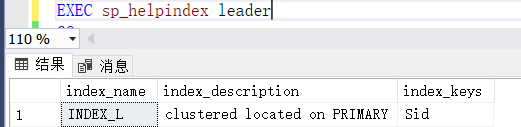


图6.5 查看索引

（2）在openGauss中创建查看索引

在命令行窗口，首先通过\c clubsystem切换数据库，再通过\c - dbtest切换用户，通过\di查看所有的index，找到了index\_l如图6.6所示

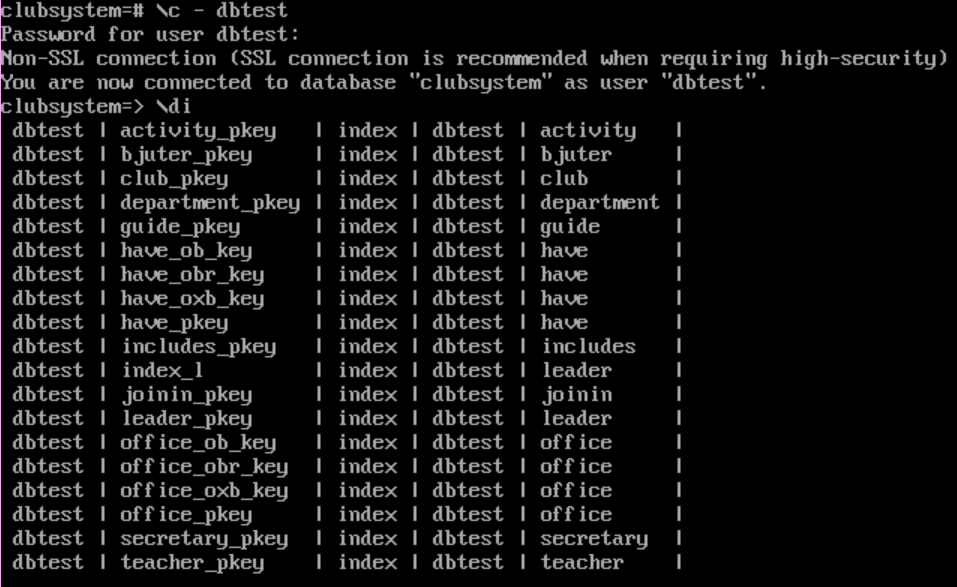


图6.6 看看你的索引

2.4 重新生成索引

（1）在SQL server中重新生成索引

可以通过重新组织索引或重新生成索引来修复索引碎片，结果如图6.7所示

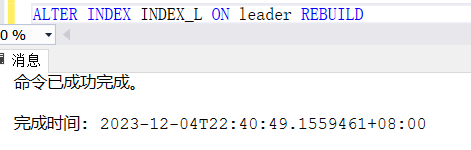


图6.7 重生成索引

（2）在openGauss中重新生成索引

在openGauss中重新生成索引的操作略有不同，详见图6.8

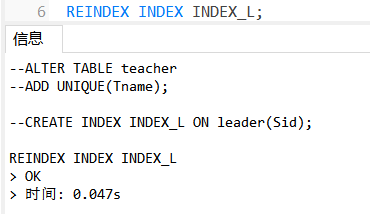


图6.8 重生成索引

2.5 删除索引

（1）在SQL server中删除索引

如图6.9所示

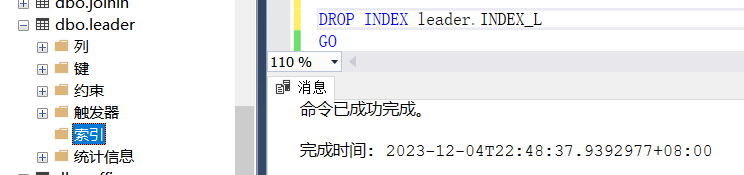


图6.9 删除成功

（2）在openGauss中删除索引

openGauss删除索引不需要添加表名，直接删除索引即可，如图6.10

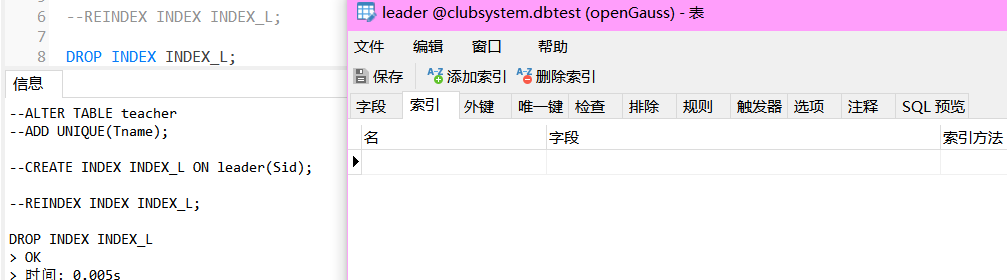
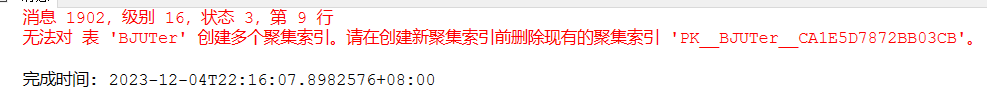


图6.10 删除成功

##### 实验中出现的问题及分析

1. 试图创建聚集索引的时候发生错误

如图6.所示，这是因为在实验二中建表时，已经为BJUTer设置了primary key唯一键，唯一键，系统生成了对应的聚集索引，应当先删除聚集索引才能创建



##### 实验中未解决的问题

本实验暂无未解决的问题

##### 思考题

Q1：索引在数据库中的作用是什么？

A1：数据库索引是一种数据结构，用于提高数据库的查询效率和性能。它通过为表中的一列或多列创建索引，使得在查询时可以快速地定位和访问所需的数据行，从而加速查询速度

Q2：索引有哪几种类型？

A2：从SQL server官方文档上来看，共有12种：哈希、内存优化非聚集索引、群集、非聚集、独特、列存储、带有包含列的索引、计算列上的索引、Filtered、空间、XML、全文

##### 心得体会

通过本实验，我掌握了索引的创建、删除和查看，以及重建操作及其意义。作为一个表的标识，索引确保了其数据的唯一性，还能帮助用户快速查询检索

#### 总结与体会

本学期《数据库原理》的数据库设计以及实验，非常有效地帮助我记忆并理解了课堂知识，通过切身的实践和不断反思，化抽象知识为具象。从一开始数据库的创建，到数据管理，再到视图和索引的初探索，我在出错和纠错的过程中，将概念付诸于实践。我感谢数据库设计和实验，帮助我对课堂知识进行了强化记忆。

通过SQL server和openGauss的双重实验，我体会到了两款数据库的不同，出去代码书写的一些不同，在视图和索引操作时，我发现了openGauss在“高安全”方面的建树，我十分赞同openGauss针对于安全性做出的工作。

其实早在今年6月份，我就参加了学校的华为基础软件训练营，当时恰好所有的考试全都结束，小学期也结束了，在军训前宝贵的休息时间和训练营之间，我选择了后者。通过两天，共十个多小时的学习，我学到了openGauss非常多的知识，从其开发背景、特性、特色，到手动实践……包括本次实验，我也用的是当时训练营华为老师给的openEuler操作系统，里面包含了openGauss，后来通过学习，我还顺利通过了openGauss OGCA初级认证。我认为，openGauss作为一款自主开发的、世界顶流水平的开源数据库，不仅展现的是中国工程师的强大水平，更展现着中国人敢于创新突破，乐于分享，美美与共天下大同的品德。

祝愿openGauss越做越好，也祝愿中国的科学家们再创辉煌！