

数据库完整性与安全性实验

【实验五】



目录

[一． 数据库完整性与安全性实验 1](#_Toc514497649)

[1． 实验目的 1](#_Toc514497650)

[2． 实验环境 1](#_Toc514497651)

[3． 实验内容与要求 2](#_Toc514497652)

[4． 实验步骤及结果分析 2](#_Toc514497653)

[5． 实验小结 10](#_Toc514497654)

2018-5-30

[裴子祥 计科七班 学号2015211921]

[指导老师：杜军平]

# 数据库完整性与安全性实验

1. 实验目的

1. 通过对完整性规则的定义实现，熟悉了解mysql中完整性保证的规则和实现方法，加深对数据完整性的理解。

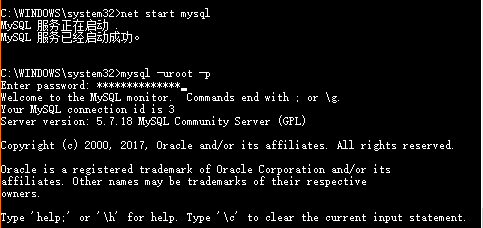
2. 通过对安全性相关内容的定义，熟悉了解mysql中安全性的内容和实现方法，加深对数据库安全性的理解。

1. 实验环境

**Microsoft Windows 10 专业版 64位**



**数据库版本：5.7.18 MySQL Community Server (GPL)**



1. 实验内容与要求
2. **完整性实验**

（1） 分别定义学生数据库中各基表的主键、外键，实现实体完整性约束和参照完整性约束；

（2） 分别向学生表、课程表插入具有相同学号和相同课程编号的学生数据和课程数据，验证其实体完整性约束；

（3） 向学生选课表中插入一条数据，课程编号是课程表中没有的，验证参照完整性约束；

（4） 删除学生表中的所有数据，验证参照完整性约束；

（5） 定义存储过程，完成查询某个学生的选课情况，并执行。

（6） 定义触发器，当向学生表插入新的一条记录时，将所有学生出生日期加1；并对其进行测试。

（7） 用sql完成以上操作。

**2. 安全性实验**

（1） 定义一新的登陆帐号、数据库用户，并授予其访问学生数据库的读权限；

（2） 分别用sa用户和新定义的用户访问学生数据库，并对其中的学生表数据进行修改；

（3） 再次用此用户访问学生数据库，并对其中的学生表数据进行修改。

（4） 用SQL语句分别完成以上内容。

1. 实验步骤及结果分析

**1、完整性实验**

（1）分别定义学生数据库中各基表的主键、外键，实现实体完整性约束和参照完整性约束：

**主键：**建表时已创建, 实现实体完整性约束。

**CREATE** **TABLE** 'student' **(**

'sno' **varchar(**6**)** **NOT** **NULL,** /\*学生学号\*/

'sname' **varchar(**8**)** **DEFAULT** **NULL,** /\*学生姓名\*/

'sex' **varchar(**2**)** **DEFAULT** **NULL,** /\*性别\*/

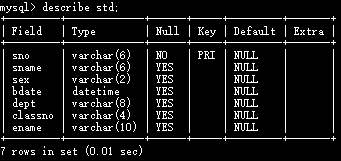
'bdate' **datetime** **DEFAULT** **NULL,** /\*生日\*/

'dept' **varchar(**8**)** **DEFAULT** **NULL,** /\*专业\*/

'classno' **varchar(**4**)** **DEFAULT** **NULL,** /\*班级\*/

**PRIMARY** **KEY** **(**'sno'**)** /\*主键为学生学号\*/

**)** ENGINE**=**InnoDB **DEFAULT** CHARSET**=**utf8**;** /\*使用utf-8编码\*/



**CREATE** **TABLE** `course` **(**

`cno` **varchar(**3**)** **NOT** **NULL,** /\*课程编号\*/

`cname` **varchar(**12**)** **DEFAULT** **NULL,** /\*课程名\*/

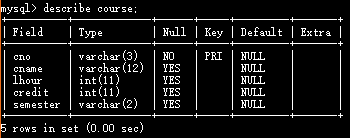
`lhour` **int(**11**)** **DEFAULT** **NULL,** /\*课程时长\*/

`credit` **int(**11**)** **DEFAULT** **NULL,** /\*学分\*/

`semester` **varchar(**2**)** **DEFAULT** **NULL,** /\*所在学期\*/

**PRIMARY** **KEY** **(**`cno`**)** /\*主键为课程编号\*/

**)** ENGINE**=**InnoDB **DEFAULT** CHARSET**=**utf8**;**



**CREATE** **TABLE** `sc` **(**

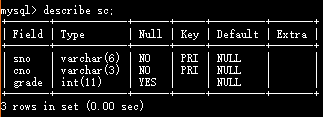
`sno` **varchar(**6**)** **NOT** **NULL,** /\*学生学号\*/

`cno` **varchar(**3**)** **NOT** **NULL,** /\*课程编号\*/

`grade` **int(**11**)** **DEFAULT** **NULL,** /\*成绩\*/

**PRIMARY** **KEY** **(**`sno`**,**`cno`**)** /\*主键为 学生学号,课程编号\*/

**)** ENGINE**=**InnoDB **DEFAULT** CHARSET**=**utf8**;**



**外键：**实现参照完整性约束。

给sc表添加两个外键

alter table sc add constraint

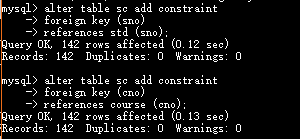
foreign key (sno)

references std (sno);

alter table sc add constraint

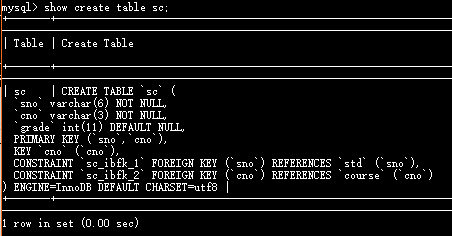
foreign key (cno)

references course (cno);

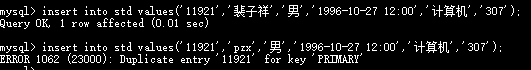


sc表发生改变，如下:

show create table sc;



（2）分别向学生表、课程表插入具有相同学号和相同课程编号的学生数据和课程数据，验证其实体完整性约束：





实体完整性约束验证成功:主键内容不能同名，所以无法插入。

（3）向学生选课表中插入一条数据，课程编号是课程表中没有的，验证参照完整性约束：



由于没有’C00’这个课程号，不能插入，参照完整性验证成功。

（4）删除学生表中的所有数据，验证参照完整性约束：



由于参照完整性约束，删除hui2破坏sc表的参照性约束，删除失败。

（5）定义存储过程，完成查询某个学生的选课情况，并执行：

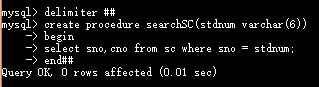
delimiter ##

create procedure searchSC(stdnum varchar(6))

begin

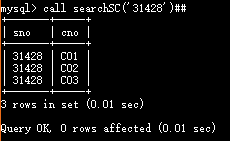
select sno,cno from sc where sno = stdnum;

end##



Delimiter ##

让命令行识别到##时才解析执行，而不是识别到第一个分号。



调用该存储过程，可以看到，将学生学号作为输入，输出该学生的选课情况。

（6）定义触发器，当向学生表插入新的一条记录时，将所有学生出生日期加1；并对其进行测试：

create table new Std select \* from std;

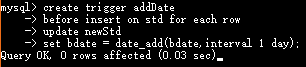


create trigger addDate

before insert on std for each row

update newStd

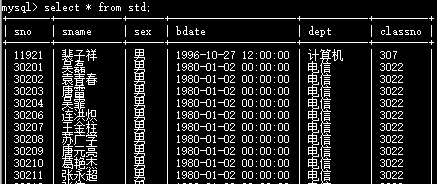
set bdate = date\_add(bdate,interval 1 day);

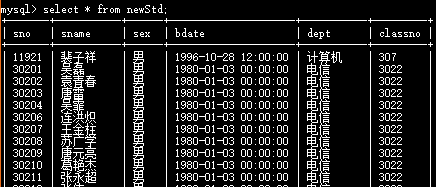


测试：









查看std与newStd表项，std中成功插入“小明”这个元组，并且触发器成功出发，更新newStd中所有表项的出生日期+1 day。

**2、安全性实验**

（1）定义一新的登陆帐号、数据库用户，并授予其访问学生数据库的读权限：

定义一个新的登入账号、数据库用户’peipeizx’，密码’123456’







对peipeizx赋予访问学生数据库的读权限

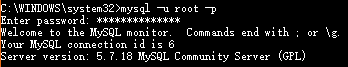


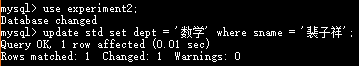
结果如下：



（2）分别用超级用户和新定义的用户访问学生数据库，并对其中的学生表数据进行修改：

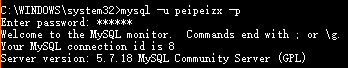
超级用户登录：





修改成功。

用户peipeizx登录：





读取学生表内容成功。



当想要修改std表内容，命令不被允许，权限不足，失败。

1. 实验小结

**实验结果：**

1. 实体完整性约束，主键能够标识表中每条信息的唯一性，不可以重复。因此，在插入与已有元组主键相同的元组时，会报错。

2. 参照完整性约束，如果插入的元组的外键不在被参照的表里，那么会报错。

3. 定义存储过程，调用后，能成功执行调用相应过程，实现查找。

4. 建立触发器后，相应可以被触发，更新表项。

5. 添加新用户后，该用户只能执行授予权限内的操作，不能执行没有授予权限的操作。

**问题记录：**

1. 在定义存储过程时，需要使用delimiter命令。delimiter是分割符的意思，因为MySQL默认以";"为分隔符。声明分割符，防止把存储过程当成SQL语句进行处理，存储过程的编译过程会报错。

2. 在建立触发器时，出现错误。MySQL对触发器的支持还没有像SQL Server那么充分，不能再原表上操作，需要创建新表来进行触发操作。

**心得总结：**

这次实验，我更加深入地理解了实体完整性约束和外键的参照完整性约束，如何定义存储过程，如何建立触发器，如何添加新用户以及更改他的操作权限等sql语句。尽管在实验过程有一些问题，但都通过查阅资料得以解决。这次实验不仅让我在实践中巩固了课堂所学信息到的理论，并且切身体会到数据库的完整性和安全性设计。