

实验五 存储过程与触发器

【实验目的】

掌握存储过程与触发器的相关操作，理解存储过程与触发器的作用。

【实验内容】

1.在实验一的基础上完成以下操作：

(1) (存储过程) 创建显示学生总人数的存储过程 STU_COUNT。

```
set serveroutput on;

create or replace procedure STU_COUNT
as
    stunum number;
begin
    select count(*) into stunum from STUDENT;

    dbms_output.put_line('学生总人数:'||stunum);
end;
/

exec STU_COUNT;
```

```
PL/SQL 过程已成功完成。
学生总人数:9
```

(2) (存储过程) 创建显示学生信息的存储过程 STUDENT_LIST。

```
create or replace procedure STUDENT_LIST
as
    CURSOR STUDENT_INFO IS SELECT SNO,SNAME,SDEPT,SCLASS,SSEX,SAGE FROM STUDENT;
begin
    FOR i IN STUDENT_INFO LOOP
        dbms_output.put_line(i.SNO||'--'||i.SNAME||'--'||i.SDEPT||'--'||i.SCLASS||'--'
        '||i.SSEX||'--'||i.SAGE);
    END LOOP;

    STU_COUNT();
END;
/

exec STUDENT_LIST;
```

```
PL/SQL 过程已成功完成。
96001--马小燕--CS--01--女--23
96002--黎明--CS--01--男--20
96003--刘东明--MA--01--男--18
96004--赵志勇--IS--02--男--20
97001--马蓉--MA--02--女--19
97002--李成功--CS--01--男--22
97003--黎明--IS--03--女--19
97004--李丽--CS--02--女--21
96005--司马志明--CS--02--男--20
学生总人数:9
```

注意不要雷同

banban

<https://github.com/dream4789/Computer-learning-resources.git>

(3) (存储过程) 创建显示某个学生平均成绩的存储过程 PRO_AVG。

```
create or replace procedure PRO_AVG ( stuno in STUDENT.SNO%type )
as
    stuavg number(4,1);
begin
    select avg(SCORE) into stuavg from SCORE where SNO = stuno;
    dbms_output.put_line('学号为:'||stuno||'的平均成绩是: '||stuavg);
END;

/

exec PRO_AVG('96001');
```

PL/SQL 过程已成功完成。
学号为:96001的平均成绩是: 83.6

(4) (存储过程) 创建显示所有学生平均成绩的存储过程 ALL_AVG。

```
create or replace procedure PRO_AVG ( stuno in STUDENT.SNO%type )
as
    stuavg number(4,1);
begin
    select avg(SCORE) into stuavg from SCORE where SNO = stuno;
    dbms_output.put_line('学号为:' || stuno || '的平均成绩是: ' || stuavg);
END;

/

exec PRO_AVG('96001');
```

[illegible]

(5) (存储过程) 创建对学生姓名进行模糊查找的过程 PRO NAME。

```
create or replace procedure PRO_NAME ( s_name in STUDENT.SNAME%type )
as
    cursor like_name is select SNAME from STUDENT where SNAME like '%'||s_name||'%';
begin
    for i in like_name loop
        dbms_output.put_line('带'||s_name||'字的姓名有: '||i.SNAME);
    end loop;
END;
```

注意不要雷同

banban

<https://github.com/dream4789/Computer-learning-resources.git>

```
/
exec PRO_NAME('李');
```

```
PL/SQL 过程已成功完成。
带 '李' 字的姓名有: 李成功
带 '李' 字的姓名有: 李丽
```

2. 根据图书借阅关系，假设存在

图书表 Book (bookID: 图书编号, bookName: 图书名, states: 状态, 状态有“借出”和“在馆”两种)

学生表 Student (stuID: 学生号, stuName: 学生名)

借阅表 BookLend (stuID, bookID, lendDate: 借书日期, returnDate: 还书日期)

当学生借书时，即在借阅表中插入一条新的记录（图书表中借出一本图书），同时把图书表中的该图书状态更改为“借出”

当学生还书时，即更新借阅表中的归还日期时（即图书归还了），同时把图书表中的该图书状态更改为“在馆”，使用触发器实现，根据提示完成下列程序。

第 1 步 创建 Book、Student 和 BookLend 三张表，并插入数据

(1) 向 Book 表插入一条记录('111','JAVA 程序设计','在馆')

```
create table BOOK
(
    bookID char(3) primary key,
    bookName varchar2(20),
    states varchar2(10),
    constraint check_states check(states in('借出','在馆'))
);
select * from BOOK;
insert into BOOK values('111','JAVA 程序设计','在馆');
```

(2) 向 Student 表插入一条记录('169074264','韩书')

```
create table STUDENT1
(
    stuID CHAR(9) primary key,
    stuName varchar2(10)
);
select * from STUDENT1;
insert into STUDENT1 values('169074264','韩书');
```

第 2 步 创建借书触发器 LEND_TRIG，当 BookLend 表中插入一条记录时，修改图书表中该图书的状态为“借出”

(3) CREATE OR REPLACE TRIGGER LEND_TRIG

```
CREATE OR REPLACE TRIGGER LEND_TRIG
after insert on BOOKLEND
for each row
```

注意不要雷同

banban

<https://github.com/dream4789/Computer-learning-resources.git>

```

begin
    update BOOK set states = '借出' where bookID = :new.bookID;
end;
/

```

第 3 步 创建还书触发器 RETURN_TRIG，当修改 BookLend 表中的归还时期时，修改图书表中该图书的状态为“在馆”。

(4) CREATE OR REPLACE TRIGGER RETURN_TRIG

```

CREATE OR REPLACE TRIGGER RETURN_TRIG
after update on BOOKLEND
for each row
begin
    update BOOK set states = '在馆' where bookID = :old.bookID;
    dbms_output.put_line(:old.stuID || ' 同学归还了 ' || :old.bookID || ' 书');
end;
/

```

第 4 步 执行以下操作，验证触发器，观察数据库中数据的变化

(5) '韩书'同学借出了'JAVA 程序设计'书，即向 BookLend 表中插入一条记录，此时 lendDate 取当前系统日期，returnDate 取空值；

```
insert into BOOKLEND values('169074264', '111', sysdate, null);
```

1行已插入。

BOOKID	BOOKNAME	STATES
1 111	JAVA程序设计	借出

STUID	BOOKID	LENDDATE	RETURNDATE
1 169074264	111	07-4月 -22	(null)

(6) '韩书'同学还了'JAVA 程序设计'书，即修改 BookLend 表中的还书日期，此时 returnDate 取当前系统日期；

```
update BOOKLEND set RETURNDATE = sysdate where stuID = '169074264';
```

1行已更新。

169074264同学归还了111书

BOOKID	BOOKNAME	STATES
1 111	JAVA程序设计	在馆

STUID	BOOKID	LENDDATE	RETURNDATE
1 169074264	111	07-4月 -22	07-4月 -22

附录：

----- 图书借阅 -----

```
create table BOOK
(
    bookID char(3) primary key,
    bookName varchar2(20),
    states varchar2(10),
    constraint check_states check(states in('借出','在馆'))
);

select * from BOOK;
insert into BOOK values('111','JAVA 程序设计','在馆');

create table STUDENT1
(
    stuID CHAR(9) primary key,
    stuName varchar2(10)
);

select * from STUDENT1;
insert into STUDENT1 values('169074264','韩书');

create table BOOKLEND
(
    stuID CHAR(9) REFERENCES STUDENT1(stuID),
    bookID CHAR(3) REFERENCES BOOK(bookID),
    lendDate date,
    returnDate date,
    CONSTRAINT book_stu primary key(stuID, bookID)
);

select * from BOOKLEND;

CREATE OR REPLACE TRIGGER LEND_TRIG
after insert on BOOKLEND
for each row
begin
    update BOOK set states = '借出' where bookID = :new.bookID;
end;

/

select * from BOOK;
select * from STUDENT1;
select * from BOOKLEND;
```

注意不要雷同

banban

<https://github.com/dream4789/Computer-learning-resources.git>

```
insert into BOOKLEND values('169074264', '111', sysdate, null);

CREATE OR REPLACE TRIGGER RETURN_TRIG
after update on BOOKLEND
for each row
begin
    update BOOK set states = '在馆' where bookID = :old.bookID;
    dbms_output.put_line(:old.stuID || ' 同学归还了 ' || :old.bookID || '书');
end;
/

update BOOKLEND set RETURNDATE = sysdate where stuID = '169074264';
delete from BOOKLEND where stuID = '169074264';
```