# ASSIGNMENT NO: 05 PL/SQL CODE BLOCK

Unnamed PL/SQL code block: Use of Control structure and Exception handling is mandatory. Write a PL/SQL block of code for the following requirements:-

Schema:

1. Borrower(Rollin, Name, DateofIssue, NameofBook, Status)
2. Fine(Roll\_no,Date,Amt)

* Accept roll\_no & name of book from user.
* Check the number of days (from date of issue), if days are between 15 to 30 then fine amount will be Rs 5per day.
* If no. of days>30, per day fine will be Rs 50 per day & for days less than 30, Rs. 5 per day.
* After submitting the book, status will change from I to R.
* If condition of fine is true, then details will be stored into fine table.

mysql> create table borrower (rollin int, name varchar (20), dateofissue date, nameofbook varchar (30), status varchar (10));

Query OK, 0 rows affected (0.45 sec)

mysql> alter table borrower drop rollin, add rollno int;

Query OK, 0 rows affected (0.23 sec) Records: 0 Duplicates: 0 Warnings: 0

mysql> describe borrower;

+-------------+------------+------+-----+---------+------+

| Field | Type | Null | Key | Default |Extra |

+-------------+------------+------+-----+---------+------+

|  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| | | name | | | varchar(20)| | YES | | | | | NULL | | |  | | |  |
| | | dateofissue | | | date| YES | |  | | | | | NULL |  | | |  | | |
| | | nameofbook | | | varchar(30)| | YES | | | | | NULL | | |  | | |  |

| status |varchar(10) | YES | | NULL | |

| rollno | int(11) | YES | | NULL | |

+-------------+------------+------+-----+---------+------+

5 rows in set (0.35 sec) mysql>

mysql> create table fine (rollno int, date date, amt decimal (5,2));

Query OK, 0 rows affected (0.14 sec)

mysql>

-- --- PL /SQL Code to define a procedure doissuebook

-- -- Four parameters are required while calling this procedure:

-- -- p1 – rollno integer

-- -- p2 – name of the borrower

-- -- p3 – date of issue

-- -- p4 – name of book

-- -- syntax for calling doissuebook:

-- -- mysql>call doissuebook(p1,p2,p3,p4);

-- -- e.g. call doissuebook(10,’PMK’ ,’01/06/2017’ ,’OS’);

-- -- usage: call doissuebook(rollno, name, 'dd/mm/yyyy'

,nameofbook); delimiter //

CREATE PROCEDURE doissuebook(p1 INT, p2 varchar(30), p3 varchar(11), p4 varchar (30))

X: BEGIN

SET @p3val=str\_to\_date(p3,'%d/%m/%Y'); SET @p5val = "I";

-- EXCEPTION HANDLING

set @errormsg = ''; if p1 <= 0 then begin

set @errormsg = 'WRONG ROLL NUMBER SPECIFIED'; select @errormsg;

leave X;

end;

end if;

if length(p4) <= 0 then begin

set @errormsg = 'WRONG BOOK SPECIFIED'; select @errormsg;

leave X;

end;

end if;

-- EXCEPTION HANDLING ENDS

insert into borrower (name, dateofissue, nameofbook, status, rollno) values (p2,@p3val,p4,@p5val,p1);

END

// delimiter ;

delimiter //

-- usage: call doreturn(rollno, bookname)

CREATE PROCEDURE doreturn(rno INT, bookname varchar(30)) BEGIN

SET @numdays=0;

SET @fineamt = 0.0; SET @numrec = 0;

select datediff(curdate(),dateofissue) from borrower

where rollno = rno and nameofbook = bookname into @numdays;

select count(\*) from borrower where rollno = rno and nameofbook = bookname into @numrec;

if @numdays >= 15 and @numdays <=30 then set @fineamt

= (@numdays-15) \* 5;

elseif @numdays > 30 then set @fineamt = ((15\*5) + (@numdays-30) \* 50);

else set @fineamt = 0; end if;

if @numrec > 0 then

update borrower set status = 'R' where rollno = rno and nameofbook = bookname;

END

//

insert into fine (rollno, date, amt) values(rno,curdate(), @fineamt);

end if;

delimiter;

+++++++++++++++++++++

mysql> call doissuebook (0,'Mrunal','2017-08-18','OPERATING SYSTEM');

+-----------------------------+

| @errormsg |

+-----------------------------+

| WRONG ROLL NUMBER SPECIFIED |

+-----------------------------+

1 row in set (0.00 sec)

Query OK, 0 rows affected, 1 warning (0.01 sec)

mysql> call doissuebook (5,'PMK','2017-06-19','OS’); Query OK, 1 rows affected, 1 warning (0.01 sec)

mysql> call doissuebook (5,'Vijay','2017-07-10','DBMS'); Query OK, 1 rows affected, 1 warning (0.01 sec)

mysql> call doissuebook (25,'Pankaj','2017-07-20','ES'); Query OK, 1 rows affected, 1 warning (0.01 sec)

mysql> call doissuebook (25,'Pankaj','2017-07-11','SQL'); Query OK, 1 rows affected, 1 warning (0.01 sec)

mysql> select \* from borrower;

+--------+-------------+------------+-------+--------+

| name | dateofissue | nameofbook | status| rollno |

+--------+-------------+------------+-------+--------+

|  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| | | PMK | | | 2017-06-19 | | | OS | | | I | | | 5 | | |
| | | PMK | | | 2017-07-10 | | | DBMS | | | I | | | 5 | | |
| | | Pankaj | | | 2017-07-20 | | | ES | | | I | | | 25 | | |
| | | Pankaj | | | 2017-07-11 | | | SQL | | | I | | | 25 | | |

+--------+-------------+------------+-------+--------+

4 rows in set (0.00 sec) mysql> call doreturn(5,'OS');

Query OK, 1 rows affected, 1 warning (0.01 sec)

mysql> call doreturn(5,'DBMS');

Query OK, 1 rows affected, 1 warning (0.01 sec)

mysql> call doreturn(25,'ES');

Query OK, 1 rows affected, 1 warning (0.01 sec)

mysql> call doreturn(25,'SQL');

Query OK, 1 rows affected, 1 warning (0.01 sec)

mysql> select \* from fine;

+--------+------------+--------+

| rollno | date | amt |

+--------+------------+--------+

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| | | 5 | | | 2017-07-30 | | | 625.00 | | |
| | | 5 | | | 2017-07-30 | | | 25.00 | | |
| | | 25 | | | 2017-07-30 | | | 0.00 | | |
| | | 25 | | | 2017-07-30 | | | 20.00 | | |

+--------+------------+--------+

4 rows in set (0.00 sec)

mysql> select \* from borrower;

+-------+-------------+------------+-------+--------+

| name | dateofissue | nameofbook | status| rollno |

+-------+-------------+------------+-------+--------+

|  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| | | PMK | | 2017-06-19 | | | OS | | | R | | | 5 | | |
| | | PMK | | 2017-07-10 | | | DBMS | | | R | | | 5 | | |
| | | Pankaj| | 2017-07-20 | | | ES | | | R | | | 25 | | |
| | | Pankaj| | 2017-07-11 | | | SQL | | | R | | | 25 | | |

+-------+-------------+------------+-------+--------+

4 rows in set (0.00 sec)

# ASSIGNMENT NO: 06 CURSOR

Cursors: (All types: Implicit, Explicit, Cursor FOR Loop, Parameterized Cursor)

Write a PL/SQL block of code using parameterized Cursor that will merge the data available in the newly created table N\_RollCall with the data available in the table O\_RollCall. If the data in the first table already exist in the second table then that data should be skipped.

COMMANDS TO CONTROL OPEN , FETCH , CLOSE

OPEN initialize the cursor with open like a sequential file

FETCH to retrieve the first row, you can execute fetch repeatedly until all rows have been retrieved.

----- Retrieve current tuple from the CURSOR to the PL/SQL CLOSE release the curor

Syntax declaration of cursor

CURSOR <CRSOR NAME> IS <SELECT STATEMENT> ;

Cursor *c1* is select empno, ename from emp; Open *c1*

Fetch *c1* into <list of variable> Fetch *c1* into <rowtype var> close C1;

end;

**MySQL**

## Database where this procedure and tables are created lab6--> use lab6;

Database changed

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\***1. Table STUDENT1 Creation** \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

MariaDB [lab6]> select \* from STUDENT1;

+------+------+------+

| id | name | age |

+------+------+------+

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| | | 1 | | | tpv | | | 37 | | |
| | | 2 | | | hab | | | 38 | | |
| | | 3 | | | nsp | | | 28 | | |

+------+------+------+

3 rows in set (0.00 sec)

\*\*\*\*\*1. **Table STUDENT2 Creation with other variable name like sid,sname,sage**\*\*\*\*\*\*\*\*\*\*

MariaDB [lab6]> select \* from STUDENT2;

+------+---------+------+

| sid | sname | sage |

+------+---------+------+

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| | | 4 | | | pragnay | | | 22 | | |
| | | 5 | | | Trisha | | | 22 | | |
| | | 6 | | | try | | | 24 | | |

+------+---------+------+

3 rows in set (0.00 sec)

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\***Procedure Creation**\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

CREATE PROCEDURE lab6\_cpy() BEGIN

DECLARE g\_id INT DEFAULT 0;

DECLARE v\_fn varchar(100); DECLARE v\_ln INTEGER DEFAULT 0; DECLARE g\_id2 INT DEFAULT 0;

DECLARE v\_fn2 varchar(100); DECLARE v\_ln2 INTEGER DEFAULT 0; DECLARE done INT DEFAULT FALSE; DECLARE Not\_Dup INT DEFAULT 0;

declare tcursor cursor FOR select distinct id,name,age from STUDENT1;

declare scursor cursor for select distinct sid,sname,sage from STUDENT2;

DECLARE CONTINUE HANDLER FOR NOT FOUND SET done = TRUE;

OPEN scursor; read\_loop: LOOP

FETCH scursor into g\_id2,v\_fn2,v\_ln2; if done then

LEAVE read\_loop;

END IF;

OPEN tcursor; read\_loopchk: LOOP

FETCH tcursor into g\_id,v\_fn,v\_ln; if done then

LEAVE read\_loopchk;

END IF;

if g\_id = g\_id2 THEN

SET NOT\_DUP=1;

LEAVE read\_loopchk;

END IF;

END LOOP read\_loopchk; IF NOT\_DUP =1 THEN

SET NOT\_DUP=0;

ELSE

insert into STUDENT1 values(g\_id2,v\_fn2,v\_ln2);

END IF;

CLOSE tcursor; SET done = FALSE;

END LOOP read\_loop; CLOSE scursor;

END

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\***calling of Procedure**\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

MariaDB [lab6]> call lab6\_cpy();

Query OK, 0 rows affected, 1 warning (0.06 sec)

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\***after Procedure check table STUDENT1\***\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

MariaDB [lab6]> select \* from STUDENT1;

+------+---------+------+

| id | name | age |

+------+---------+------+

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| | | 1 | | | tpv | | | 37 | | |
| | | 2 | | | hab | | | 38 | | |
| | | 3 | | | nsp | | | 28 | | |
| | | 4 | | | pragnay | | | 22 | | |
| | | 5 | | | Trisha | | | 22 | | |
| | | 6 | | | try | | | 24 | | |

+------+---------+------+

6 rows in set (0.00 sec) MariaDB [lab6]>

# ORACLE

SQL>select \* from emp1;

|  |  |  |
| --- | --- | --- |
| **EMPID** | **ENAME1** | **SAL1** |
| 3 | ttt | 3000 |
| 4 | latest | 500 |
| 1 | aaa | 2222 |
| 2 | try | 3000 |

SQL> select \* from emp2; no data found

**CREATE CURSOR**

Declare

Cursor empcur is

Select empid, ename1, sal1 from emp1; a emp1.empid %type;

b emp1.ename1 %type; c emp1.sal1 %type;

x number; begin

loop

open empcur;

fetch empcur into a,b,c; exit when empcur % notfound;

select count(\*) into x from emp2 where sr=a; if x=0 then

insert into emp2 values (a,b,c); end if;

end loop; close empcur; end;

> select \* from emp2;

|  |  |  |
| --- | --- | --- |
| **SR** | **NAME** | **INCOME** |
| 3 | ttt | 3000 |
| 4 | latest | 500 |
| 1 | aaa | 2222 |
| 2 | try | 3000 |

# ASSIGNMENT NO: 07 PROCEDURE & FUNCTION

Write a Stored Procedure namely proc\_Grade for the categorization of student. If marks scored by students in examination is <=1500 and marks>=990 then student will be placed in distinction category if marks scored are between 989 and900 category is first class, if marks 899 and 825 category is Higher Second Class

Write a PL/SQL block for using procedure created with above requirement. Stud\_Marks(rollno, name, total\_marks) Result(Roll,Name, Class)

SQL> create table stud\_marks(Roll\_no number(10), Name varchar2(20) , Marks number(5));

SQL> desc stud\_marks;

|  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| **Table** | **Colum n** | **Data Type** | **Lengt h** | **Precisi on** | **Scal e** | **Primary Key** | **Nullab le** | **Defau lt** | **C** | **omme nt** |
| STUD\_MAR KS | ROLL\_ NO | Number | - | 10 | 0 | - | - | | - | |
|  | NAME | Varchar2 | 20 | - | - | - | - | | - | |
|  | MARKS | Number | - | 5 | 0 | - | - | | - | |

SQL> select \* from stud\_marks;

|  |  |  |
| --- | --- | --- |
| **ROLL\_ NO** | **NAME** | **MARKS** |
| 101 | ABC | 1000 |
| 102 | XYZ | 900 |
| 103 | UVW | 850 |
| 104 | PQR | 800 |

SQL> create table Result(Roll\_no number(10), Name varchar2(20), Result varchar2(25));

SQL> desc Result;

|  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| **Table** | **Colum n** | **Data Type** | **Lengt h** | **Precisio n** | **Scal e** | **Primary Key** | **Nullabl e** | **Defau lt** | **C** | **omme nt** |
| RESUL T | ROLL\_N O | Number | - | 10 | 0 | - | - | | - | |
|  | NAME | Varchar2 | 20 | - | - | - | - | | - | |
|  | RESULT | Varchar2 | 25 | - | - | - | - | | - | |

SQL> select \* from Result; no data found

**CREATE PROCEDURE**

SQL> create or replace procedure grade(temp IN number, P\_Roll\_No OUT stud\_marks.Roll\_No%type,

P\_Name OUT stud\_marks.Name%type, P\_Total OUT stud\_marks.Marks%type) AS begin

select Roll\_No,Name,Marks into P\_Roll\_No,P\_Name,P\_Total from stud\_marks where Roll\_No=temp;

if P\_Total <= 1500 and p\_Total >= 990 then

insert into Result values (P\_Roll\_No, P\_Name,'DISTINCTION'); else if

P\_Total <= 989 and p\_Total >= 900 then

insert into Result values (P\_Roll\_No, P\_Name,'FIRST

CLASS');

else if

P\_Total <= 899 and p\_Total >= 825 then insert into Result values (P\_Roll\_No,

P\_Name,'HIGHER SECOND CLASS');

else

insert into Result values (P\_Roll\_No,

P\_Name,'FAIL');

end if;

end if;

end if; exception

when no\_data\_found then

dbms\_output.put\_line (' Roll Number'|| temp || ' NOT FOUND'); end;

**Calling Procedure**

SQL> declare temp number(20);

P\_Roll\_No stud\_marks.Roll\_No%type; P\_Name stud\_marks.Name%type; P\_Total stud\_marks.Marks%type; begin

temp:=&temp; grade(temp,P\_Roll\_No,P\_Name,P\_Total); end;

Enter value for temp: 101 Old 7: temp := & temp; New 7: temp := 101;

**CREATE FUNCTION**

SQL> create or replace function fun\_grade(temp IN number) Return number AS

P\_Roll\_No stud\_marks.Roll\_No%type; P\_Name stud\_marks.Name%type; P\_Total stud\_marks.Marks%type;

begin

select Roll\_No,Name,Marks into P\_Roll\_No,P\_Name,P\_Total from stud\_marks where Roll\_No=temp;

if P\_Total <= 1500 and p\_Total >= 990 then

insert into Result values (P\_Roll\_No, P\_Name,'DISTINCTION'); else if

P\_Total <= 989 and p\_Total >= 900 then

insert into Result values (P\_Roll\_No, P\_Name,'FIRST

CLASS');

else if

P\_Total <= 899 and p\_Total >= 825 then insert into Result values (P\_Roll\_No,

P\_Name,'HIGHER SECOND CLASS');

else

insert into Result values (P\_Roll\_No,

P\_Name,'FAIL');

end if;

end if;

end if;

return P\_Roll\_No; exception

when no\_data\_found then

dbms\_output.put\_line (' Roll Number'|| temp || ' NOT FOUND'); end;

**Calling Function**

SQL> declare

temp number(20):=&temp; P\_Roll\_No varchar2(20); begin

P\_Roll\_No := fun\_grade(temp); end;

# ASSIGNMENT NO: 08 TRIGGER

Database Trigger (All Types: Row level and Statement level triggers, Before and After Triggers). Write a database trigger on Library table. The System should keep track of the records that are being updated or deleted. The old value of updated or deleted records should be added in Library\_Audit table. Frame the problem statement for writing Database Triggers of all types.



\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*create schema1 Borrower\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\* SQL> create table borrower (rollno number(5), name varchar2(20), dateofissue date, book varchar2(30), status varchar2(5));

SQL> desc borrower;

Name Null? Type

----------------------------------------- -------- ----------------------------

ROLLNO NUMBER(5)

NAME VARCHAR2(20)

DATEOFISSUE DATE

BOOK VARCHAR2(30)

STATUS VARCHAR2(5)

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*create schema1 Library\_Audit\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\* SQL> create table library\_audit(rollno number(5), name varchar2(20), dateofissue date, book varchar2(30), status varchar2(5),ts timestamp);

SQL> desc library\_audit;

Name Null? Type

----------------------------------------- -------- ---------------------------- ROLLNO NUMBER(5)

NAME VARCHAR2(20)

DATEOFISSUE DATE

BOOK VARCHAR2(30)

STATUS VARCHAR2(5)

TS TIMESTAMP(6)

SQL> select \* from borrower;

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **ROLLNO** | **NAME** | **DATEOFISSUE** | **BOOK** | **STATUS** |
| 101 | abc | 16-JUL-17 | dbms | r |
| 102 | abc1 | 16-JUL-17 | cn | i |
| 103 | abc2 | 18-JUL-17 | toc | i |
| 104 | abc3 | 20-JUL-17 | ds | i |

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\***Trigger AFTER INSERT AND UPDATE**\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\* create or replace trigger lib\_trigger after insert or update on borrower for each row when(new.rollno!=0)

begin

dbms\_output.put\_line ('new number'||:new.rollno); insert into library\_audit

values(:new.rollno,:new.name,:new.dateofissue,:new.book,:new.stat us,current\_timestamp);

end;

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\***Trigger Before INSERT and UPDATE**\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\* create or replace trigger lib\_trigger before insert or update on borrower for each row when(new.rollno!=0)

begin

dbms\_output.put\_line ('new number'||:new.rollno); insert into library\_audit

values(:new.rollno,:new.name,:new.dateofissue,:new.book,:new.stat us,current\_timestamp);

end;

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\***Table BORROWER after Insertion**\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\* SQL> select \* from borrower;

ROLLNO NAME DATEOFISS BOOK STATU

---------- -------------------- --------- ------------------------------ ------------

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 101 | abc | 16-JUL-17 | dbms | r |
| 102 | abc1 | 16-JUL-17 | cn | i |
| 103 | abc2 | 18-JUL-17 | toc | i |
| 104 | abc3 | 20-JUL-17 | ds | i |
| 105 | abc43 | 23-JUL-17 | daa | r |
| 106 | xyz | 10-AUG-17 | sp | r |
| 0 | ll | 12-AUG-17 | lj | i |
| 107 | ll | 12-AUG-17 | dbms | i |
| 0 | ll | 12-AUG-17 | lj | i |

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\***Table LIBRARY\_AUDIT after Insertion**\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

SQL> update borrower set book='dbms' where rollno=107; new number107

1 row updated.

SQL> select \* from library\_audit;

ROLLNO NAME DATEOFISS BOOK STATUS TS

----------------------------------------------------------------------------------------------------------------------------

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 107 ll  107 ll | 12-AUG-17  12-AUG-17 | lj  dba | I  I | 23-AUG-17 02.53.53.014000 PM  23-AUG-17 02.54.48.887000 PM |
|  |  |  |  |  |

107 ll 12-AUG-17 dbms I 23-AUG-17 03.04.11.491000 PM

SQL> select \* from borrower;

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\***DELETE Trigger On BORROWER**\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\* create or replace trigger lib\_trigger before delete on borrower for each row when(old.rollno!=0)

begin

dbms\_output.put\_line ('new number'||:old.rollno); insert into library\_audit

values(:old.rollno,:old.name,:old.dateofissue,:old.book,:old.stat us,current\_timestamp);

end;

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*TABLES AFTER DELETE OPERATION\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

SQL> select \* from borrower;

ROLLNO NAME DATEOFISS BOOK STATUS

---------- -------------------- --------- ------------------------------ ------------------------

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 102 | abc1 | 16-JUL-17 | cn | i |
| 103 | abc2 | 18-JUL-17 | toc | i |
| 106 | xyz | 10-AUG-17 | sp | r |
| 0 | ll | 12-AUG-17 | lj | i |
| 0 | ll | 12-AUG-17 | lj | i |

SQL> delete from borrower where rollno=102; new number102

1. row deleted.

SQL> select \* from library\_audit;

ROLLNO NAME DATEOFISS BOOK STATUS TS

--------------------------------------------------------------------------------------------------------------------------------------- 102 abc1 16-JUL-17 cn I 23-AUG-17 03.24.35.023000 PM

SQL> delete from borrower where rollno=0;

1. rows deleted.

SQL> select \* from library\_audit;

ROLLNO NAME DATEOFISS BOOK STATU TS

---------------------------------------------------------------------------------------------------------------------------------------- 102 abc1 16-JUL-17 cn I 23-AUG-17 03.24.35.023000 PM

SQL>