MANNAR THIRUMALAI NAICKER COLLEGE (AUTONOMOUS)
PASUMALAI, MADURAI-625 004.

DEPARTMENT OF COMPUTER SCIENCE

LAB RECORD

B.Sc COMPUTER SCIENCE

Academic Year 2022-2023

III Year V Semester

Relational Database Management System – Lab

(18UCSCP5)

MANNAR THIRUMALAI NAICKER COLLEGE (AUTONOMOUS)

PASUMALAI, MADURAI-04.

DEPARTMENT OF COMPUTER SCIENCE

BONAFIDE CERTIFICATE

NAME	**:**
REGISTER NUMBER	**:**
CLASS	**: III B.Sc(CS) 'A'**
COURSE NAME WITH CODE	**: Relational Database Management System – Lab (18UCSCP5)**

This is to certify that record is a bonafide work done by the above mentioned student. This certificate is awarded for the same.

COURSE IN-CHARGE	**HEAD OF THE DEPARTMENT**
Mr.M.Selvakumar.,M.Sc.,M.Phil.,	Dr.G.Devika.,MCA.,M.Phil.,PhD.,

Submitted for practical examination held on ……………. at Mannar Thirumalai Naicker College(Autonomous), Pasumalai, Madurai.

INTERNAL EXAMINER **EXTERNAL EXAMINER**

Table of Content

EX. NO: 01 **DDL COMMANDS**

DATE:

AIM:

To implement the DDL Commands in SQL.

COMMANDS:

```
SQL> create table employee(empid varchar(10),empname varchar(15),gender
varchar(10);
Table created.

SQL> desc employee;
 Name                          Null?          Type
 ---------------------------   ------------   -----------------
 EMPID                                        VARCHAR(10)
 EMPNAME                                      VARCHAR(15)
 GENDER                                       VARCHAR(10)


SQL> insert into employee values ('&empid', '&empname,'&gender');
Enter value for empid: 1001
Enter value for empname:Lingesh
Enter value for gender: male

old 1: insert into student1 values ('&empid', '&empname,'&gender') new 1:
insert into student1 values ('1001', 'Lingesh','male')
1 row created.
```

```
SQL> insert into employee values ('&empid', '&empname,'&gender');
Enter value for empid: 1002
Enter value for empname:Sundar
Enter value for gender: male

old 1: insert into student1 values ('&empid', '&empname,'&gender')

new 1: insert into student1 values ('1002', 'Sundar','male')
1 row created.

SQL> insert into employee values ('&empid', '&empname,'&gender');
Enter value for empid: 1003
Enter value for empname:Aathi
Enter value for gender: male

old 1: insert into student1 values ('&empid', '&empname,'&gender') new 1:
insert into student1 values ('1003', 'Aathi','male')
1 row created.

SQL> insert into employee values ('&empid', '&empname,'&gender');
Enter value for empid: 1004
Enter value for empname:Karthi
Enter value for gender: male

old 1: insert into student1 values ('&empid', '&empname,'&gender') new 1:
insert into student1 values ('1004', 'Karthi','male')
1 row created.


SQL> select * from employee;
EMPID     EMPNAME      GENDER
---------  -----------   ---------------
1001        Lingesh       male
1002        Sundar        male
1003        Aathi         male
1004        Karthi        male
```

```
SQL> alter table employee add(empage number(3));
 Table altered.

SQL> desc employee;
 Name                          Null?          Type
 --------------------------   ------------   -----------------
 EMPID                                        VARCHAR(10)
 EMPNAME                                      VARCHAR(15)
 GENDER                                       VARCHAR(10)
 EMPAGE                                       NUMBER(3)

SQL> truncate table student1;
Table truncated.

SQL> select * from student1;
no rows selected
```

RESULT:

Thus the above all DDL Commands are successfully executed and the output is verified

EX NO :02 **DML COMMANDS**

DATE :

AIM:

To implement the DML commands using sql.

COMMANDS:

```
SQL>create table studentmarks (stdid varchar(10),branch varchar(15),m1 number(3),m2 number(3),m3
number(3),m4 number(3),m5 number(5),total number(3));
Table Created.
SQL>insert into studentmarks values(' &stdid', '&branch ',&m1,&m2,&m3,&m4,&m5,&total);
Enter value for stdid =S001
Enter value for branch=IT
Enter value for m1=90
Enter value for m2=80
Enter value for m3=100
Enter value for m4=75
Enter value for m5=95
Enter value for total=440
1 row created.
SQL>/
Enter value for stdid =S002
Enter value for branch=IT
Enter value for m1=80
Enter value for m2=70
Enter value for m3=90
Enter value for m4=100
Enter value for m5=90
Enter value for total=430
1 row created.
SQL>/
Enter value for stdid =S003
Enter value for branch=CS
Enter value for m1=95
Enter value for m2=75
Enter value for m3=100
Enter value for m4=80
Enter value for m5=90
Enter value for total=440
1 row created.
SQL>select * from studentmarks;

STDI        BRANC        M1        M2        M3        M4
D M5        H
            TOTAL

S001        IT
95          440          90        80        100       75

S002        IT
90          430          80        70        90        100

S003        CS
90          440          95        75        100       80
```

```
SQL> update studentmarks set branch='IT' where stdid='S003';

STDI        BRANC        M1        M2        M3        M4
D M5        H
            TOTAL

S001        IT
95          440          90        80        100       75

S002        IT
90          430          80        70        90        100

S003        IT
90          440          95        75        100       80
SQL>delete from studentmarks where stdid='S003';
1 row deleted.
```

RESULT:

Thus the above program was successfully executed and the output is verified.

EX NO:03 **Program to Demonstrate Queries**
DATE:

AIM:

To implement Queries using Hospital Data in sql.

COMMANDS:

```
SQl>create table Hospital (docid varchar (10), docname varchar(20), department varchar (25),
Qualification varchar (15), experience number (2));

Table created

SQL> insert into Hospital values (‘&docid’, '&docname',’&
   department','&Qualification',&experience);

Enter values for docid: D001
Enter values for doc name :Lingesh
Enter values for department: cardiology
Enter values for Qualification : MBBS
Enter values for experience : 10
1 row Created
SQL>/
Enter values for docid: D002
Enter values for doc name :Sundar
Enter values for Qualification : MBBS
Enter values for experience : 4
1 row Created
SQL>/
Enter values for docid: D003
Enter values for doc name :Palpandi
Enter values for department: Skin
Enter values for Qualification : MD
Enter values for experience : 4
1 row Created
```

```
SQL> select * from Hospital;
 DOCID     DOCNAME          DEPARTMENT            QUALIFICATION
-----------------         --------------          ---------
EXPERIENCE
------
D001      Lingesh          cardiology                    MBBS
       10
D002      Sundar           Orthopedics                   MBBS
        4

D003      Palpandi         Skin                          MD
        4

SQL> select docname, department, Qualification from hospital where experience>='5'
and Qualification !='MD';

DOCNAME           DEPARTMENT            QUALIFICATION
---------------------------             ---------
Lingesh           cardiology                MBBS

SQL>select docname,experience from hospital where department='skin';
DOCNAME           EXPERIENCE
---------------------------
Palpandi            4

SQL> update Hospital set experience='5' where docid='D003';
1 row updated.

SQL> delete from Hospital where docid='D003';
1 row deleted.
```

RESULT :

Thus the above all Hospital Queries are successfully executed and the output is verified.

EX NO :04 **NESTED SUB QUERIES**

DATE:

AIM:

To implement the Nested Queries using sql.

COMMANDS:

```
SQL>create table emp_det(eno number(3),ename varchar2(20),address varchar2(30),basicsal
number(12,5),Dno number(3));
Table Created.
SQL>create table prodetails(pno number(3),pname varchar(10),noofstaff number(3));
Table Created.
SQL>create table workin(pno number(3),eno number(3),pjob varchar(12));
Table Created.
SQL>insert into emp_det values(&eno, '&ename', '&address',&basicsal,&dno);
Enter values for eno:101
Enter values for ename:Lingesh
Enter values for address:Madurai
Enter values for basicsal:25000
Enter values for dno:10
1 row created.
SQL>/
Enter values for eno:102
Enter values for ename:Sundar
Enter values for address:Madurai
Enter values for basicsal:23000
Enter values for dno:10
1 row created.
SQL>/
Enter values for eno:103
Enter values for ename:Karthi
Enter values for address:Dindgul
Enter values for basicsal:20000
Enter values for dno:2
1 row created.
```

```
SQL>insert into prodetails values('&pno ', '&pname', &noofstaff);
Enter values for pno:1
Enter values for pname:cloud computing
Enter values for noofstaff:2
1 row created.
SQL>/
Enter values for pno:2
Enter values for pname:compiler
Enter values for noofstaff:5
1 row created.
SQL>/
Enter values for pno:3
Enter values for pname:AI
Enter values for noofstaff:1
1 row created.
SQL>insert into workin values(&pno,&eno, '&pjob');
Enter values for pno:1
Enter values for eno:101
Enter values for pjob:programmer
1 row created.
SQL>/
Enter values for pno:2
Enter values for eno:102
Enter values for pjob:analyst
1 row created.
SQL>/
Enter values for pno:3
Enter values for eno:103
Enter values for pjob:datamanager
1 row created.
```

```
SQL>select * from emp_det;
ENO   ENAME          ADDRESS          BASICSAL           DNO
-------  ----------        -----------------        -----------------          ------
101     Lingesh          Madurai            25000                 10
102     Sundar           Madurai             23000                10
103     Karthi           Dindgul              20000                 2

SQL>select * from prodetails ;
PNO   PNAME              NOOFSTAFF
----------------------------------------------
1       cloud computing           2
2       compiler                        5
3       AI                                 1
SQL>select * from workin;
PNO   ENO                    PJOB
-----   -----------------       -------------
1       101                      programmer
2       102                      analyst
3       103                      datamanager

SQL> select ename from emp_det where dno not in (select dno from emp_det where
ename='Karthi');
ENAME
Lingesh
Sundar

SQL> select ename,dno from emp_det where dno = (select dno from emp_det where
ename='Sundar');
   ENAME                                               DNO
   Lingesh                                             10
   Sundar                                              10
```

```
SQL>select ename,basicsal from emp_det where basicsal > (select min(basicsal) from emp_det
where dno=10 order by ename);
```

```
ENAME                    BASICSAL
Lingesh                  25000
```

```
SQL> select pno,pname from prodetails where exists (select pno from workin where
workin.pno=prodetails.pno);
```

```
PNO              PNAME
1                cloud computing
2                compiler
3                AI
```

RESULT :

Thus the above all Nested Queries are successfully executed and the output is verified.

EX NO:05 AGGREGATE FUNCTIONS

DATE :

AIM:

To implement the aggregate functions using sql.

COMMANDS:

```
SQL> create table emp8(emid number(5),emname varchar(20),emsal number(10),emadd

varchar(20));

Table created.

SQL> desc emp8;

 Name                             Null?              Type

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

 EMID                                                NUMBER(5)

 EMNAME                                              VARCHAR2(20)

 EMSAL                                               NUMBER(10)

 EMADD                                               VARCHAR2(20)

SQL> insert into emp8 values(&emid,'&emname',&emsal,'&emadd');
Enter value for emid: 1
Enter value for emname: alagar
Enter value for emsal: 20000
Enter value for emadd: madurai
old 1: insert into emp8 values(&emid,'&emname',&emsal,'&emadd')
new 1: insert into emp8 values(1,'alagar',20000,'madurai')
1 row created.

SQL> /
Enter value for emid: 2
Enter value for emname: selva
Enter value for emsal: 10000
Enter value for emadd: viluppuram
old 1: insert into emp8 values(&emid,'&emname',&emsal,'&emadd')
new 1: insert into emp8 values(2,'selva',10000,'viluppuram')
1 row created.
```

```
SQL> /
Enter value for emid: 3
Enter value for emname: kumar
Enter value for emsal: 15000
Enter value for emadd: chennai
old 1: insert into emp8 values(&emid,'&emname',&emsal,'&emadd')
new 1: insert into emp8 values(3,'kumar',15000,'chennai')

1 row created.

SQL>/
Enter value for emname: guru
Enter value for emsal: 50000
Enter value for emadd: apk
old 1: insert into emp8 values(&emid,'&emname',&emsal,'&emadd')
new 1: insert into emp8 values(4,'guru',50000,'apk')
1 row created.

SQL> /
Enter value for emid: 5
Enter value for emname:arul
Enter value for emsal: 9000
Enter value for emadd: kpt
old 1: insert into emp8 values(&emid,'&emname',&emsal,'&emadd')
new 1: insert into emp8 values(5,'arul',9000,'kpt')
1 row created.

SQL> select * from emp8;
      EMID        EMNAME            EMSAL      EMADD

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

       1          alagar            20000       madurai
       2          selva             10000       viluppuram
       3          kumar             15000       chennai
       4          guru              50000       apk
       5          arul              9000        kpt
```

```
SQL> select count(emsal) from emp8;
COUNT(EMSAL)
---------------------------
          5

SQL> select sum(emsal) from emp8;
SUM(EMSAL)
-----------------------

    104000

SQL> select avg(emsal) from emp8;
AVG(EMSAL
----------------------
     20800

SQL> select min(emsal) from emp8;
MIN(EMSAL)
----------------------

      9000

SQL> select max(emsal) from emp8;
MAX(EMSAL)
-----------------------

     50000
```

RESULT :

Thus the above all aggregate functions are successfully executed and the output is verified.

EX.NO: 06 **RELATIONAL ALGEBRA OPERATIONS**

DATE :

AIM:

To implement the Relational Algebra Operation using sql.

COMMANDS:

SQL> create table library5(bid number(10),bname varchar(20),bauthname varchar(20),brelyr number(8));

Table created.

SQL> desc library5;

```
Name                    Null?                Type
-------------           --------------       ------------------
BID                                          NUMBER(10)
BNAME                                        VARCHAR2(20)
BAUTHNAME                                    VARCHAR2(20)
BRELYR                                       NUMBER(8)
```

SQL> insert into library5 values(&bid,'&bname','&bauthname',&brelyr);

Enter value for bid: 1

Enter value for bname: illakkiyam

Enter value for bauthname: selva

Enter value for brelyr: 2000

old 1: insert into library5 values(&bid,'&bname','&bauthname',&brelyr)

new 1: insert into library5 values(1,'illakkiyam','selva',2000)

1 row created.

```
SQL> /

Enter value for bid: 2

Enter value for bname: kethai

Enter value for bauthname: alagar

Enter value for brelyr: 1998

old 1: insert into library5 values(&bid,'&bname','&bauthname',&brelyr)

new 1: insert into library5 values(2,'kethai','alagar',1998)

1 row created.

SQL> /

Enter value for bid: 3

Enter value for bname: vinayaga

Enter value for bauthname: murugan

Enter value for brelyr: 1990

old 1: insert into library5 values(&bid,'&bname','&bauthname',&brelyr)

new 1: insert into library5 values(3,'vinayaga','murugan',1990)

1 row created.

SQL> /
Enter value for bid: 4

Enter value for bname: tamil

Enter value for bauthname: arun

Enter value for brelyr: 2002

old 1: insert into library5 values(&bid,'&bname','&bauthname',&brelyr)

new 1: insert into library5 values(4,'tamil','arun',2002)

1 row created.
```

```
SQL> /

Enter value for bid: 5

Enter value for bname: science

Enter value for bauthname: guru

Enter value for brelyr: 2005

old 1: insert into library5 values(&bid,'&bname','&bauthname',&brelyr)

new 1: insert into library5 values(5,'science','guru',2005)

1 row created.

SQL> create table study3(bid number(10),bname varchar(20));

Table created.

SQL> desc study3;

 Name                      Null?                    Type

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

 BID                                                 NUMBER(10)

 BNAME                                               VARCHAR2(20)

SQL> insert into study3 values(&bid,'&bname');

Enter value for bid: 2

Enter value for bname: kethai

old 1: insert into study3 values(&bid,'&bname')

new 1: insert into study3 values(2,'kethai')

1 row created.
```

```
SQL> /
Enter value for bid: 3
Enter value for bname: vinayaga
old 1: insert into study3 values(&bid,'&bname')
new 1: insert into study3 values(3,'vinayaga')
1 row created.
SQL> /
Enter value for bid: 4
Enter value for bname: tamil
old 1: insert into study3 values(&bid,'&bname')
new 1: insert into study3 values(4,'tamil')
1 row created.
SQL> /
Enter value for bid: 6
Enter value for bname: kadahi
old 1: insert into study3 values(&bid,'&bname')
new 1: insert into study3 values(6,'kadahi')
1 row created.
SQL> /
Enter value for bid: 7
Enter value for bname: maths
old 1: insert into study3 values(&bid,'&bname')
new 1: insert into study3 values(7,'maths')
1 row created.
```

```
SQL> select * from study3;

      BID        BNAME

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

       2          kethai

       3         vinayaga

       4          tamil

       6         kadahi

       7         maths

SQL> select * from library5;

   BID          BNAME              BAUTHNAME              BRELYR

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

       1        illakkiyam             selva                   2000

       2        kethai                 alagar                  1998

       3        vinayaga               murugan                 1990

       4         tamil                 arun                    2002

       5        science                guru                    2005
```

```
SQL> select bname from library5 union select bname from study3;

BNAME

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

illakkiyam
kadahi
kethai
maths
science
tamil
vinayaga

7 rows selected.


SQL> select bname from library5 union all select bname from study3;

BNAME

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

illakkiyam

kethai

vinayaga

tamil

science

kethai

vinayaga

tamil

kadahi

maths

10 rows selected.
```

```
SQL> select bname from library5 minus select bname from study3;

BNAME

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

illakkiyam

science

SQL> select bname from library5 intersect select bname from study3;

BNAME

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

kethai

tamil

vinayaga
```

RESULT:

Thus the above all relational algebra operations are successfully executed and the output is verified.

EX.NO: 07 **IMPLEMENTATION OF JOINS**

DATE :

AIM:

To implement the different joins using sql.

COMMAND:

```
SQL> create table books1(bookname varchar(20),bookid number(10),bookautname
varchar(15),bookprize number(12));

Table created.

SQL> desc books1;

 Name                    Null?             Type

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

 BOOKNAME                                  VARCHAR2(20)

 BOOKID                                    NUMBER(10)

 BOOKAUTNAME                               VARCHAR2(15)

 BOOKPRIZE                                 NUMBER(12)

SQL> insert into books1 values('&bookname',&bookid,'&bookautname',&bookprize);

Enter value for bookname: gk

Enter value for bookid: 1

Enter value for bookautname: bharath

Enter value for bookprize: 50

old 1: insert into books1 values('&bookname',&bookid,'&bookautname',&bookprize)

new 1: insert into books1 values('gk',1,'bharath',50)

1 row created.
```

```
SQL> /

Enter value for bookname: malar

Enter value for bookid: 2

Enter value for bookautname: raju

Enter value for bookprize: 60

old 1: insert into books1 values('&bookname',&bookid,'&bookautname',&bookprize)

new 1: insert into books1 values('malar',2,'raju',60)

1 row created.

SQL> /

Enter value for bookname: time

Enter value for bookid: 3

Enter value for bookautname: louis

Enter value for bookprize: 40

old 1: insert into books1 values('&bookname',&bookid,'&bookautname',&bookprize)

new 1: insert into books1 values('time',3,'louis',40)

1 row created.

SQL> create table library0(bookname varchar(20),bookid number(12),bookautname
varchar(15),bookreleyr number(10));

Table created.
```

```
SQL> desc library0;

 Name                     Null?                Type

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

 BOOKNAME                                       VARCHAR2(20)

 BOOKID                                         NUMBER(12)

 BOOKAUTNAME                                    VARCHAR2(15)

 BOOKRELEYR                                     NUMBER(10)


SQL> insert into library0 values('&bookname',&bookid,'&bookautname',&bookreleyr);

Enter value for bookname: gk

Enter value for bookid: 1

Enter value for bookautname: bharath

Enter value for bookreleyr: 2002

old 1: insert into library0 values('&bookname',&bookid,'&bookautname',&bookreleyr)

new 1: insert into library0 values('gk',1,'bharath',2002)

1 row created.
```

```
SQL> /

Enter value for bookname: eng

Enter value for bookid: 4

Enter value for bookautname: mani

Enter value for bookreleyr: 2003

old 1: insert into library0 values('&bookname',&bookid,'&bookautname',&bookreleyr)

new 1: insert into library0 values('eng',4,'mani',2003)

1 row created.

SQL> /

Enter value for bookname: phy

Enter value for bookid: 6

Enter value for bookautname: sankar

Enter value for bookreleyr: 2006

old 1: insert into library0 values('&bookname',&bookid,'&bookautname',&bookreleyr)

new 1: insert into library0 values('phy',6,'sankar',2006)

1 row created.

SQL> select * from books1;

BOOKNAME              BOOKID          BOOKAUTNAME           BOOKPRIZE

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

gk                        1               bharath                  50

malar                     2               raju                     60

time                      3               louis                    40
```

```
SQL> select * from library0;

BOOKNAME        BOOKID       BOOKAUTNAME        BOOKRELEYR
-----------    --------     ---------------    -------------
eng               4             mani               2002
tamil             5             bala               2007
phy               6             sankar             2006
gk                1             bharath            2002
eng               4             mani               2003
phy               6             sankar             2006
6 rows selected.

SQL> select books1.bookname,library0.bookautname from books1 inner join library0 on
books1.bookid=library0.bookid;

BOOKNAME       BOOKAUTNAME
-----------   ----------------
gk                  bharath

SQL> select books1.bookname,library0.bookautname from books1 left join library0 on
books1.bookid=library0.bookid;

BOOKNAME       BOOKAUTNAME
----------    -------------
gk                bharath
time
malar
```

SQL> select books1.bookname,library0.bookautname from books1 right join library0 on books1.bookid=library0.bookid;

```
BOOKNAME          BOOKAUTNAME

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

gk                               bharath

                                 sankar

                                 sankar

                                 mani

                                 bala

                                 mani
```

6 rows selected.

SQL> select books1.bookname,library0.bookautname from books1 full outer join library0 on books1.bookid=library0.bookid;

```
BOOKNAME          BOOKAUTNAME

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

gk                                bharath

malar                             sankar

                                   sankar

                                   mani

                                   bala

                                   mani
```

8 rows selected

RESULT:

Thus the above queries are successfully executed and the output is verified.

EX.NO:08 **FIBONACCI SERIES**

DATE :

AIM:

To write a PL/SQL program for generate the Fibonacci series of the given input.

CODING:

```
SQL> Set Serveroutput on
declare
first number:=0;
second number:=1;
third number;
n number:=&n;
i number;
begin
dbms_output.put_line('Fibonacci series is:');
dbms_output.put_line(first);
dbms_output.put_line(second);
 for i in 2..n
loop
third:=first+second;
first:=second;
second:=third;
dbms_output.put_line(third);
end loop;
end;
/
```

Output:

SQL>

```
Enter value for n: 6

old 5: n number:=&n;

new 5: n number:=6;

Fibonacci series is:

0

1

1

2

3

5

8
```

RESULT:

Thus the above program was successfully executed and the output is verified.

EXNO:09 **FACTORIAL CALCULATION**

DATE :

AIM:

To write a PL/SQL program for find the factorial value of the given number.

CODING:

```
SQL> declare
n number(3);
i number(3):=1;
f number(3):=1;
begin
n:=&n;
dbms_output.put_line('no is:'||n);
while(i<=n)
loop
f:=f*i;
i:=i+1;
end loop;
dbms_output.put_line('factorial number is:'||f);
end;
/
```

```
SQL> Set serveroutput on

SQL > /

Enter value for n: 5

old 6: n:=&n;

new 6: n:=5;

PL/SQL procedure successfully completed.

SQL> set serveroutput on

SQL> /

Enter value for n: 5

old 6: n:=&n;

new 6: n:=5;

no is:5

factorial number is:120

PL/SQL procedure successfully completed.
```

RESULT:

Thus the above program was successfully executed and the output is verified.

EX NO:10 **ODD OR EVEN NUMBERCHECKING**

DATE :

AIM:

To write a PL/SQL program for find the given number is odd or even.

CODING:

```
SQL> declare
n  number(3);
begin
n:=&n;
dbms_output.put_line('no is:'||n);
if(n mod 2=0)then
dbms_output.put_line('even');
else
dbms_output.put_line('odd');
end if;
end;
/
```

```
SQL> set serveroutput on

SQL> /

Enter value for n: 6

old 4: n:=&n;

new 4: n:=6;

no is:6

even

PL/SQL procedure successfully completed.


SQL> /

Enter value for n: 7

old 4: n:=&n;

new 4: n:=7;

no is:7

odd

PL/SQL procedure successfully completed.
```

RESULT:

Thus the above program was successfully executed and the output is verified.

EX NO:11 **Program To Demonstrate Exception Handling**

DATE :

AIM:

To write a PL/SQL program to execute the exception handling process**.**

CODING:

```
SQL> Create Table Customers (
            Id Int Not Null,
            Name Varchar (20) Not Null,
            Age Int Not Null,
            Address Char (25),
            Salary Decimal (18, 2),
            Primary Key (Id)
        );

SQL > Table Created

SQL> insert into customers values ( &id,'&name', &age,'&address',&salaray);
Enter value for id: 101
Enter value for name: hasen
Enter value for age: 21
Enter value for address: madurai
Enter value for salaray: 15000
old 1: insert into customers values ( &id,'&name', &age,'&address',&salaray)
new 1: insert into customers values ( 101,'hasen', 21,'madurai',15000)

1 row created.

SQL> /
Enter value for id: 102
Enter value for name: komal
Enter value for age: 22
Enter value for address: MP
Enter value for salaray: 4500
old 1: insert into customers values ( &id,'&name', &age,'&address',&salaray)
new 1: insert into customers values ( 102,'komal', 22,'MP',4500)

1 row created.

SQL> /
Enter value for id: 103
Enter value for name: khilan
Enter value for age: 25
Enter value for address: delhi
Enter value for salaray: 20000
old 1: insert into customers values ( &id,'&name', &age,'&address',&salaray)
new 1: insert into customers values ( 103,'khilan', 25,'delhi',20000)

1 row created.

DECLARE
       c_id customers.id%type := 8;
    c_name customerS.Name%type;
    c_addr customers.address%type;
    BEGIN
       SELECT name, address INTO c_name, c_addr
       FROM customers
       WHERE id = c_id;
       DBMS_OUTPUT.PUT_LINE ('Name: '|| c_name);
      DBMS_OUTPUT.PUT_LINE ('Address: ' || c_addr);

   EXCEPTION
      WHEN no_data_found THEN
         dbms_output.put_line('No such customer!');
      WHEN others THEN
         dbms_output.put_line('Error!');
   END;
   /
No such customer!

PL/SQL procedure successfully completed.
```

Raising Exceptions:

```
Set serveroutput on
SQL> DECLARE
       c_id customers.id%type := &cc_id;
       c_name customerS.Name%type;
       c_addr customers.address%type;
       -- user defined exception
       ex_invalid_id EXCEPTION;
    BEGIN
       IF c_id <= 0 THEN
          RAISE ex_invalid_id;
      ELSE
         SELECT name, address INTO c_name, c_addr
         FROM customers
         WHERE id = c_id;
         DBMS_OUTPUT.PUT_LINE ('Name: '|| c_name);
         DBMS_OUTPUT.PUT_LINE ('Address: ' || c_addr);
      END IF;

   EXCEPTION
      WHEN ex_invalid_id THEN
         dbms_output.put_line('ID must be greater than zero!');
    WHEN no_data_found THEN
      dbms_output.put_line('No such customer!');
    WHEN others THEN
      dbms_output.put_line('Error!');
  END;
  /
```

OUTPUT:

```
Enter value for cc_id: -103
old   2:    c_id customers.id%type := &cc_id;
new   2:    c_id customers.id%type := -103;
ID must be greater than zero!

PL/SQL procedure successfully completed.
```

RESULT:

Thus the above program was successfully executed and the output is verified.

EX NO:12 **PROGRAM TO IMPLMENT PROCEDURES**
DATE :

AIM:

To write a PL/SQL program to execute the Procedures.

CODING:

```
Set serveroutput on
DECLARE
  a number;
  b number;
  c number;
PROCEDURE findMin (x IN number, y IN number, z OUT number) IS
BEGIN
  IF x < y THEN
    z:= x;
  ELSE
    z:= y;
  END IF;
END;
BEGIN
  a:= 23;
  b:= 45;
  findMin(a, b, c);
  dbms_output.put_line(' Minimum of (23, 45) : ' || c);
END;
/
```

When the above code is executed at the SQL prompt, it produces the following result −

```
Minimum of (23, 45) : 23

PL/SQL procedure successfully completed.
```

RESULT:

Thus the above program was successfully executed and the output is verified.

EX NO:13 **PROGRAM TO IMPLEMENT TRIGGER**
DATE :

AIM:

To implement the Trigger using SQL.

CODING:

```
SQL> Create Table Customers (
            Id Int Not Null,
            Name Varchar (20) Not Null,
            Age Int Not Null,
            Address Char (25),
            Salary Decimal (18, 2),
            Primary Key (Id)
        );

SQL > Table Created

SQL> insert into customers values ( &id,'&name', &age,'&address',&salaray);
Enter value for id: 101
Enter value for name: hasen
Enter value for age: 21
Enter value for address: madurai
Enter value for salaray: 15000
old 1: insert into customers values ( &id,'&name', &age,'&address',&salaray)
new 1: insert into customers values ( 101,'hasen', 21,'madurai',15000)

1 row created.

SQL> /
Enter value for id: 102
Enter value for name: komal
Enter value for age: 22
Enter value for address: MP
Enter value for salaray: 4500
old 1: insert into customers values ( &id,'&name', &age,'&address',&salaray)
new 1: insert into customers values ( 102,'komal', 22,'MP',4500)

1 row created.

SQL> /
Enter value for id: 103
Enter value for name: khilan
Enter value for age: 25
Enter value for address: delhi
Enter value for salaray: 20000
```

```
old 1: insert into customers values ( &id,'&name', &age,'&address',&salaray)
new 1: insert into customers values ( 103,'khilan', 25,'delhi',20000)

1 row created.
```

Trigger Creations:

```
CREATE OR REPLACE TRIGGER display_salary_changes
    BEFORE DELETE OR INSERT OR UPDATE ON customers
    FOR EACH ROW
    WHEN (NEW.ID > 0)
    DECLARE
       sal_diff number;
    BEGIN
       sal_diff := :NEW.salary - :OLD.salary;
      dbms_output.put_line('Old salary: ' || :OLD.salary);
      dbms_output.put_line('New salary: ' || :NEW.salary);
      dbms_output.put_line('Salary difference: ' || sal_diff);
   END;
   /

Trigger created.
```

INSERTING:

```
INSERT INTO CUSTOMERS (ID,NAME,AGE,ADDRESS,SALARY)
  2 VALUES (7, 'Kriti', 22, 'HP', 7500.00 );
Old salary:
New salary: 7500
Salary difference:

1 row created.
```

UPDATING:

```
SQL> UPDATE Customers
    SET  salary=salary+1000
    WHERE id=7;

1 row updated.

SQL> select * from Customers;

       ID NAME                  AGE ADDRESS              SALARY
---------------------------------------------------- ---------
      101 hase                   21   madurai             15000
      103 khilan                 25   delhi               20000
        7  Kriti                 22   HP                   8500
```

DELETING:

```
DELETE Customers
  2 where id=102;

1 row deleted.
```

```
SQL>select * from Customers;

      ID NAME                    AGE ADDRESS                    SALARY
----------------------------------------------------------------------------
     101 hase                     21    madurai                  15000
     103 khilan                   25    delhi                    20000
       7 Kriti                    22    HP                        7500
```

RESULT:

Thus the above program was successfully executed and the output is verified.

EX NO: 14
DATE :

CURSORS

AIM:

To write a PL/SQL program to execute the Cursors.

CODING:

```
SQL> Create Table Customers (
          Id Int Not Null,
          Name Varchar (20) Not Null,
          Age Int Not Null,
          Address Char (25),
          Salary Decimal (18, 2),
          Primary Key (Id)
        );

SQL > Table Created

SQL> insert into customers values ( &id,'&name', &age,'&address',&salaray);
Enter value for id: 101
Enter value for name: hasen
Enter value for age: 21
Enter value for address: madurai
Enter value for salaray: 15000
old 1: insert into customers values ( &id,'&name', &age,'&address',&salaray)
new 1: insert into customers values ( 101,'hasen', 21,'madurai',15000)

1 row created.

SQL> /
Enter value for id: 102
Enter value for name: komal
Enter value for age: 22
Enter value for address: MP
Enter value for salaray: 4500
old 1: insert into customers values ( &id,'&name', &age,'&address',&salaray)
new 1: insert into customers values ( 102,'komal', 22,'MP',4500)

1 row created.

SQL> /
Enter value for id: 103
Enter value for name: khilan
Enter value for age: 25
Enter value for address: delhi
Enter value for salaray: 20000
old 1: insert into customers values ( &id,'&name', &age,'&address',&salaray)
new 1: insert into customers values ( 103,'khilan', 25,'delhi',20000)

1 row created.
```

The following program will update the table and increase the salary of each customer by 500 and use the **SQL%ROWCOUNT** attribute to determine the number of rows affected –

CURSOR CREATION:

```
Set serveroutput on
SQL>
SQL> DECLARE
       total_rows number(2);
    BEGIN
       UPDATE customers
       SET salary = salary + 500;
       IF sql%notfound THEN
          dbms_output.put_line('no customers selected');
       ELSIF sql%found THEN
          total_rows := sql%rowcount;
         dbms_output.put_line( total_rows || ' customers selected ');
      END IF;
   END;
   /
Old salary: 15000
New salary: 15500
Salary difference: 500
Old salary: 20000
New salary: 20500
Salary difference: 500
Old salary: 8500
New salary: 9000
Salary difference: 500
3 customers selected

PL/SQL procedure successfully completed.

SQL> select * from Customers;

       ID NAME                 AGE ADDRESS              SALARY
-------------------------------------------------------------
      101 hase                  21   madurai             15000
      103 khilan                25   delhi               20000
        7 Kriti                 22   HP                   8500
```

EXPLICIT CURSOR:

```
Set serveroutput on
SQL> DECLARE
       c_id customers.id%type;
       c_name customers.name%type;
       c_addr customers.address%type;
       CURSOR c_customers is
          SELECT id, name, address FROM customers;
    BEGIN
```

```
     OPEN c_customers;
     LOOP
    FETCH c_customers into c_id, c_name, c_addr;
      EXIT WHEN c_customers%notfound;
      dbms_output.put_line(c_id || ' ' || c_name || ' ' || c_addr);
    END LOOP;
    CLOSE c_customers;
  END;
  /
101 hase madurai
103 khilan delhi
7 Kriti HP

PL/SQL procedure successfully completed.
```

RESULT:

Thus the above program was successfully executed and the output is verified.

EX NO:15 **PACKAGES**
DATE:

AIM:
To write a PL/SQL program to execute the Packages.

CODING:

```
 CREATE OR REPLACE PACKAGE c_package AS
  -- Adds a customer
  PROCEDURE addCustomer(c_id customers.id%type,
  c_name customers.Name%type,
  c_age customers.age%type,
  c_addr customers.address%type,
  c_sal  customers.salary%type);

  -- Removes a customer
 PROCEDURE delCustomer(c_id  customers.id%TYPE);
 --Lists all customers
 PROCEDURE listCustomer;

END c_package;
/
```

```
Package created.
```

```
CREATE OR REPLACE PACKAGE BODY c_package AS
   PROCEDURE addCustomer(c_id customers.id%type,
     c_name customers.Name%type,
     c_age customers.age%type,
     c_addr customers.address%type,
     c_sal  customers.salary%type)
   IS
   BEGIN
     INSERT INTO customers (id,name,age,address,salary)
       VALUES(c_id, c_name, c_age, c_addr, c_sal);
  END addCustomer;

  PROCEDURE delCustomer(c_id customers.id%type) IS
  BEGIN
    DELETE FROM customers
    WHERE id = c_id;
  END delCustomer;

  PROCEDURE listCustomer IS
  CURSOR c_customers is
    SELECT name FROM customers;
  TYPE c_list is TABLE OF customers.Name%type;
  name_list c_list := c_list();
  counter integer :=0;
  BEGIN
```

```
         FOR n IN c_customers LOOP
         counter := counter +1;
         name_list.extend;
         name_list(counter) := n.name;
         dbms_output.put_line('Customer(' ||counter|| ')'||name_list(counter));
         END LOOP;
      END listCustomer;

   END c_package;
   /

Package body created.

set serveroutput on
SQL> DECLARE
       code customers.id%type:= 7;
    BEGIN
       c_package.addcustomer(8, 'Rajnish', 25, 'Chennai', 3500);
       c_package.addcustomer(9, 'Subham', 32, 'Delhi', 7500);
       c_package.listcustomer;
       c_package.delcustomer(code);
       c_package.listcustomer;
    END;
   /
Old salary:
New salary: 3500
Salary difference:
Old salary:
New salary: 7500
Salary difference:
Customer(1)hase
Customer(2)khilan
Customer(3)Kriti
Customer(4)Rajnish
Customer(5)Subham
Customer(1)hase
Customer(2)khilan
Customer(3)Rajnish
Customer(4)Subham

PL/SQL procedure successfully completed.
```

RESULT:

Thus the above program was successfully executed and the output is verified.