**********Experiment no:-05**********

Author:Saurabh Khandagale
Roll No:46
Date :25-September-2020

EXPERIMENT:-05

AIM:-

Establish the database relation EMPLOYEE and populate it with sample records. The logical schema of EMPLOYEE table is.

Problem Statement:

Establish the database relation EMPLOYEE and populate it with sample records. The logical schema of EMPLOYEE table is

==========Query-01==========

Write SQL code to create and execute an anonymous PL/SQL block that will insert 5 tuples into EXAM. Ensure to commit the populated records. Test the insertion in EXAM by displaying its contents.

==========

```
SQL> CREATE TABLE EXAM
       (UROLL VARCHAR(20) CHECK (UROLL>=1001 OR UROLL<=1099),
        COURSE VARCHAR2(20) CHECK  (COURSE='DBMS'),
        EXAMDT DATE DEFAULT SYSDATE +5,
        CONSTRAINT PK_UNROLL PRIMARY KEY(UROLL));

Table created.

SQL> SELECT CONSTRAINT_TYPE,CONSTRAINT_NAME,TABLE_NAME
     FROM USER_CONSTRAINTS
     WHERE TABLE_NAME='EXAM';

C CONSTRAINT_NAME               TABLE_NAME
- ------------------------------ ----------------------
C SYS_C007782                   EXAM
C SYS_C007783                   EXAM
P PK_UNROLL                     EXAM.
```

```
    DECLARE
    CURSOR C1 IS SELECT  UROLL,COURSE,EXAMDT  FROM EXAM;
    ROLL EXAM.UROLL%TYPE;
    COR EXAM.COURSE%TYPE;
    EXDT EXAM.EXAMDT%TYPE;
    BEGIN
    INSERT INTO EXAM VALUES(1001,'DBMS',SYSDATE+5);
    INSERT INTO EXAM VALUES(1002,'DBMS',SYSDATE+5);
    INSERT INTO EXAM VALUES(1003,'DBMS',SYSDATE+5);
    INSERT INTO EXAM VALUES(1004,'DBMS',SYSDATE+5);
    INSERT INTO EXAM VALUES(1005,'DBMS',SYSDATE+5);
    COMMIT;
    OPEN C1;
    LOOP
    FETCH C1 INTO ROLL,COR,EXDT;
    EXIT WHEN C1%NOTFOUND;
    DBMS_OUTPUT.PUT_LINE(ROLL||' '|| COR|| ''||EXDT  );
    END LOOP;
    CLOSE C1;
    END;
    /

OUTPUT:-
1001 DBMS 26-SEP-20
1002 DBMS 26-SEP-20
1003 DBMS 26-SEP-20
1004 DBMS 26-SEP-20
1005 DBMS 26-SEP-20


      ================================Query02=================================
      Write SQL code to create and execute an anonymous PL/SQL block that will
      use %TYPE variables to populate the EMPP table with corresponding tuples
      in EMPLOYEE table.
      ========================================================================


SQL> CREATE TABLE EMPP AS SELECT ENO AS EID,FNAME ||' '||LNAME AS
ENAME,HIREDATE AS HIREDATE,DESIGNATION AS DESIGNATION,SALARY AS SALARY FROM
EMPLOYEE WHERE 1=0;

Table created.

SQL> ALTER TABLE EMPP ADD  PRIMARY KEY (EID);

Table altered.

SQL> SELECT CONSTRAINT_TYPE,CONSTRAINT_NAME,TABLE_NAME
        FROM USER_CONSTRAINTS
         WHERE TABLE_NAME='EMPP';

C CONSTRAINT_NAME               TABLE_NAME
- ------------------------------ ------------------------------
C SYS_C007801                   EMPP
C SYS_C007802                   EMPP
C SYS_C007803                   EMPP
C SYS_C007804                   EMPP
P SYS_C007805                   EMPP
```

```
SQL> SELECT * FROM EMPP;

no rows selected

SQL> DESC EMPP;
 Name              Null?    Type
 ----------------- -------- ------------
 EID               NOT NULL NUMBER(4)
 ENAME                      VARCHAR2(21)
 HIREDATE          NOT NULL DATE
 DESIGNATION       NOT NULL VARCHAR2(15)
 SALARY            NOT NULL NUMBER(8,2)

SQL> BEGIN
    INSERT   INTO EMPP (EID,ENAME,HIREDATE,DESIGNATION,SALARY)  (SELECT
     ENO,FNAME||' '||LNAME,
     HIREDATE,DESIGNATION, SALARY FROM EMPLOYEE );
    END;
    /

PL/SQL procedure successfully completed.

SQL> SELECT * FROM EMPP;

       EID ENAME                HIREDATE  DESIGNATION         SALARY
---------- -------------------- --------- --------------- ----------
      7102 Samantha Jones       08-NOV-06 Professor           146500
      7101 Eugene Sabatini      10-OCT-06 Professor           150000
      7103 Alexander Lloyd      01-FEB-07 Professor           148000
      7104 Simon Downing        01-SEP-07 Professor           138400
      7107 Christov Plutnik     01-SEP-08 Asso. Professor     127400
      7105 Christina Mulboro    15-JUL-08 Asso. Professor     127400
      7106 Dolly Silverline     17-AUG-08 Asso. Professor     127400
      7108 Ellena Sanchez       12-NOV-09 Asso. Professor     119700
      7109 Martina Jacobson     15-NOV-09 Asst. Professor      91000
      7110 William Smithfield   23-JUN-10 Asst. Professor      86400
      7111 Albert Greenfield    12-JUL-16 Research Asst.       48200
      7112 James Washington     22-AUG-17 Research Asst.       44600
      7113 Julia Martin         01-DEC-18 Teaching Asst.       35600
      7114 Larry Gomes          18-MAY-19 Teaching Asst.       32850
      7115 Svetlana Sanders     15-JAN-20 Teaching Asst.       30000
      7116 Lovelyn Brendon      17-JUL-20 Teaching Asst.       30000
      7117 Hector Hercules      01-AUG-20 Teaching Asst.       32200

17 rows selected.
```

```
=====================================Query03============================
Write SQL code to create and execute an anonymous PL/SQL block that will use
%TYPE variables to populate the EMPP table with corresponding tuples in
EMPLOYEE table.
========================================================================

SQL> SQL> CREATE TABLE MENTEE AS SELECT SF.SID ,SF.NAME AS
STAFF_NAME,STD.FNAME ||' '||STD.LNAME AS STU_NAME,STD.ROLL,STD.REG_DT FROM
STUDENT STD,STAFF1 SF WHERE 1=0;

Table created.
SQL> DESC MENTEE;
 Name              Null?    Type
 ----------------- -------- ------------
 SID               NOT NULL NUMBER(3)
 STAFF_NAME        NOT NULL VARCHAR2(25)
 STU_NAME                   VARCHAR2(31)
 ROLL                       NUMBER(3)
 REG_DT            NOT NULL DATE



SQL> ALTER TABLE MENTEE ADD CONSTRAINT PK_SNO_ROLL PRIMARY KEY (SID,ROLL);

Table altered.

SQL> SELECT CONSTRAINT_TYPE,CONSTRAINT_NAME,TABLE_NAME
                FROM USER_CONSTRAINTS
                WHERE TABLE_NAME='MENTEE';

SQL>  SELECT CONSTRAINT_TYPE,CONSTRAINT_NAME,TABLE_NAME
                    FROM USER_CONSTRAINTS
                    WHERE TABLE_NAME='MENTEE';

C CONSTRAINT_NAME                TABLE_NAME
- ------------------------------ ------------------------------
C SYS_C007821                    MENTEE
C SYS_C007822                    MENTEE
C SYS_C007823                    MENTEE
P PK_SNO_ROLL                    MENTEE

SQL> SELECT * FROM MENTEE;

no rows selected
```

```
SQL> declare
     CURSOR C1 IS ( SELECT SF.SID,SF.NAME ,STD.FNAME||' '||STD.LNAME AS
     STU_NAME,STD.ROLL,STD.REG_DT  FROM STUDENT STD INNER JOIN STAFF SF  ON
     STD.ADVISOR=SF.SID );
     v_rec c1%ROWTYPE;
    begin
    open c1;
    loop
     fetch c1 INTO v_rec;
     exit when c1%NOTFOUND;

   --DBMS_OUTPUT.PUT_LINE(V_REC.SID);
   INSERT INTO MENTEE(SID,STAFF_NAME,STU_NAME,ROLL,REG_DT)   VALUES (
   V_REC.SID,V_REC.NAME,V_REC.STU_NAME,V_REC.ROLL,V_REC.REG_DT);
   end loop;
   close c1;
  end;
   /

PL/SQL procedure successfully completed.


SQL> SELECT * FROM MENTEE;

       SID STAFF_NAME          STU_NAME                   ROLL REG_DT
---------- ------------------- -------------------- ---------- ---------
       101 Kamalkant Marathe   Afra Sayed                 1 20-JUL-18
       104 Aasawari Deodhar    Akansha Wasalu             2 20-JUL-18
       108 Jasmine Arora       Anjali Rajendran           3 19-JUL-18
       109 Vallabh Pai         Aradhita Menghal           4 07-JUL-18

       102 Adishesh Vidyarthi  Saurabh Khandagale        46 10-AUG-19
       102 Adishesh Vidyarthi  Paritosh Dandekar         57 14-JUL-18

       109 Vallabh Pai         NAVEEN NAMJOSHI           88 14-AUG-19
       110 Harmeet Khullar     TIPNIS TUSHAR             89 14-AUG-19
       101 Kamalkant Marathe   Cinderella Goldsmith      91 18-AUG-19
       104 Aasawari Deodhar    Sebstain Ford             92 18-AUG-19

77 rows selected.
```

```
=====================================Query04============================
 Write SQL code to create and execute an anonymous PL/SQL block that will
display the contents of MENTEE table without using declared variables. You
should format the output using RPAD() and/or LPADC), while including proper
headers in the result.
========================================================================


DECLARE
TYPE  MENTEE IS RECORD
(
SD NUMBER(30),
SFAME VARCHAR(50),

SDNAME VARCHAR(35),
RL NUMBER(10),
RD DATE
);

M1 MENTEE;--OBJECT
BEGIN
DBMS_OUTPUT.PUT_LINE(RPAD('SID',5)||'  '||RPAD('SFAME',8)||'  '
||RPAD('SDNAME',10)||' '||RPAD(M1.RL,5)||' ' ||RPAD('DATE',6));
FOR I IN 1.. 19
LOOP

SELECT  SID,STAFF_NAME,STU_NAME,ROLL,REG_DT
  INTO M1.SD,M1.SFAME,M1.SDNAME,M1.RL,M1.RD FROM MENTEE  WHERE ROLL=I;

DBMS_OUTPUT.PUT_LINE(RPAD(M1.SD,5)||'  '||RPAD(M1.SFAME,8)||'  '
||RPAD(M1.SDNAME,10)||' '||RPAD(M1.RL,6)||' ' ||RPAD(M1.RD,7));

END LOOP;


FOR I IN 30.. 48
LOOP

SELECT  SID,STAFF_NAME,STU_NAME,ROLL,REG_DT
  INTO M1.SD,M1.SFAME,M1.SDNAME,M1.RL,M1.RD FROM MENTEE  WHERE ROLL=I;

DBMS_OUTPUT.PUT_LINE(RPAD(M1.SD,5)||'  '||RPAD(M1.SFAME,8)||'  '
||RPAD(M1.SDNAME,10)||' '||RPAD(M1.RL,6)||' ' ||RPAD(M1.RD,7));

END LOOP;

FOR I IN 51.. 68
LOOP

SELECT  SID,STAFF_NAME,STU_NAME,ROLL,REG_DT
  INTO M1.SD,M1.SFAME,M1.SDNAME,M1.RL,M1.RD FROM MENTEE  WHERE ROLL=I;




DBMS_OUTPUT.PUT_LINE(RPAD(M1.SD,5)||'  '||RPAD(M1.SFAME,8)||'  '
||RPAD(M1.SDNAME,10)||' '||RPAD(M1.RL,6)||' ' ||RPAD(M1.RD,7));

END LOOP;
```

```
FOR I IN 71.. 92
LOOP

SELECT  SID,STAFF_NAME,STU_NAME,ROLL,REG_DT
  INTO M1.SD,M1.SFAME,M1.SDNAME,M1.RL,M1.RD FROM MENTEE  WHERE ROLL=I;

DBMS_OUTPUT.PUT_LINE(RPAD(M1.SD,5)||'  '||RPAD(M1.SFAME,8)||'  '
||RPAD(M1.SDNAME,10)||' '||RPAD(M1.RL,6)||' ' ||RPAD(M1.RD,7));

END LOOP;

END;
/
```

OUTPUT:-

```
SID    SFAME     SDNAME             DATE
101    Kamalkan  Afra Sayed 1      20-JUL-
104    Aasawari  Akansha Wa 2      20-JUL-
108    Jasmine   Anjali Raj 3      19-JUL-
109    Vallabh   Aradhita M 4      07-JUL-
102    Adishesh  Ketki Fadn 5      14-JUL-
110    Harmeet   Lalita Sha 6      10-JUL-
103    Manishi   Muskan Gup 7      19-JUL-
106    Deo Nara  Prateeksha 8      13-JUL-
105    Geetika   Priyal Tao 9      19-JUL-
107    Sanjeev   Rashi Chou 10     08-AUG-



103    Manishi   Yash Jain  84     03-JUL-
105    Geetika   Yogesh Sir 85     21-JUL-
107    Sanjeev   Shapath Pa 86     27-JUL-
108    Jasmine   Mayank Ran 87     25-JUL-
109    Vallabh   NAVEEN NAM 88     14-AUG-
110    Harmeet   TIPNIS TUS 89     14-AUG-
```

```
==================================Query05=====================================
Write SQL code to create and execute an anonymous PL/SQL block that will
display the system date. Use exception (exception VALUE_ERROR) to check if the
variable holding the system date is large enough in size.
==============================================================================


declare
    DT varchar2(15) ;
    D1  DATE ;
 VALUE_ERROR EXCEPTION;
begin
DT:='&DT';
IF(LENGTH(DT)=11) THEN
SELECT TO_DATE(DT,'DD/MM/YYYY') INTO D1 FROM DUAL;
DBMS_OUTPUT.PUT_LINE(D1);
ELSE
RAISE VALUE_ERROR;
END IF;
EXCEPTION WHEN VALUE_ERROR THEN
DBMS_OUTPUT.PUT_LINE('*********INCORRECT DATA**********');
end;
/

Enter value for dt: 20-AUG-20000
old   6: DT:='&DT';
new   6: DT:='20-AUG-20000';
*********INCORRECT DATA**********

PL/SQL procedure successfully completed.

SQL> /
Enter value for dt: 20-AUG-2000
old   6: DT:='&DT';
new   6: DT:='20-AUG-2000';
20-AUG-00

PL/SQL procedure successfully completed.
```

```
====================================Query06================================

Write SQL code to create and execute an anonymous PL/SQL block that will check
(say, for employee number 7108) whether an employee is entitled to receive the
longevity bonus. Longevity bonus is given to employees with minimum 12 year of
service. Now, re-execute the block to extend longevity bonus to employees with
10 years of service.
================================================================================
SQL> DECLARE
      CURSOR c_emp IS select  extract(year from sysdate) - extract (year from
      hiredate) from employee;
    CURSOR C1 IS SELECT * FROM EMPLOYEE;
    TT EMPLOYEE%ROWTYPE;
    TEMP1 NUMBER(3);
    BEGIN
      OPEN c_emp;
      OPEN C1;
     dbms_output.put_line('EMPLOYEE NAME THAT ARE ELIGIBLE FOR LONGITUTED
     BOUNS {12 YEAR}');
      LOOP
       FETCH c_emp into  TEMP1;
     FETCH C1 INTO TT;


   IF(TEMP1>=12) THEN
       dbms_output.put_line(TT.FNAME||'  '||TT.LNAME);
   END IF;
       EXIT WHEN c_emp%NOTFOUND;
     END LOOP;
     CLOSE c_emp;
   END;
   /
EMPLOYEE NAME THAT ARE ELIGIBLE FOR LONGITUTED BOUNS {12 YEAR}
Samantha  Jones
Eugene  Sabatini
Alexander  Lloyd
Simon  Downing
Christov  Plutnik
Christina  Mulboro
Dolly  Silverline

PL/SQL procedure successfully completed.

SQL>

[PART 2]

DECLARE
  CURSOR c_emp IS select  extract(year from sysdate) - extract (year from
hiredate) from employee;
CURSOR C1 IS SELECT * FROM EMPLOYEE;
TT EMPLOYEE%ROWTYPE;
TEMP1 NUMBER(3);
BEGIN
  OPEN c_emp;
  OPEN C1;
 dbms_output.put_line('EMPLOYEE NAME THAT ARE ELIGIBLE FOR LONGITUTED BOUNS
{10 YEAR}');
  LOOP
```

```
   FETCH c_emp into  TEMP1;
   FETCH C1 INTO TT;


IF(TEMP1>=10) THEN
    dbms_output.put_line(TT.FNAME||'  '||TT.LNAME);
END IF;
    EXIT WHEN c_emp%NOTFOUND;
  END LOOP;
  CLOSE c_emp;
END;
/

EMPLOYEE NAME THAT ARE ELIGIBLE FOR LONGITUTED BOUNS {10 YEAR}
Samantha  Jones
Eugene  Sabatini
Alexander  Lloyd
Simon  Downing
Christov  Plutnik
Christina  Mulboro
Dolly  Silverline
Ellena  Sanchez
Martina  Jacobson
William  Smithfield

PL/SQL procedure successfully completed.

====================================Query07================================
Write SQL code to create and execute an anonymous PL/SQL block that will
locate the first August born employee. Re-write and execute an anonymous
PL/SQL block that will locate the first August born employee, when EMPLOYEE
table is searched in reversed order.
============================================================================

DECLARE
NN NUMBER(3);
N1 NUMBER(3);
T_NO NUMBER(5);
MTH NUMBER(10);
EENO EMPLOYEE.ENO%TYPE;
NAME1 EMPLOYEE.FNAME%TYPE;
NAME2 EMPLOYEE.FNAME%TYPE;
CTN NUMBER:=0;
CURSOR C1 IS SELECT ENO ,FNAME,LNAME FROM EMPLOYEE;


CURSOR  C2 IS select extract(day from BIRTHDATE)   from employee;

CURSOR  C3 IS select extract(month from BIRTHDATE)  from employee;
BEGIN
OPEN C1;
OPEN C2;
OPEN C3;

SELECT COUNT(*) INTO NN FROM EMPLOYEE;
DBMS_OUTPUT.PUT_LINE('IN NORMAL  ORDER');
FOR I IN   1.. NN
LOOP
FETCH C2 INTO  N1 ;
```

```
FETCH C3 INTO  MTH ;
 FETCH C1 INTO EENO,NAME1,NAME2;
EXIT WHEN C1%notfound;


if(N1=1) AND (MTH=8) then
CTN:=CTN+1;
DBMS_OUTPUT.PUT_LINE(EENO||'  '||NAME1||'  '||NAME2||''||I);
end if;
END LOOP;
DBMS_OUTPUT.PUT_LINE('NO. OF RECORD FOUND '||CTN);


DBMS_OUTPUT.PUT_LINE('IN REVERSE ORDER');


FOR I IN  REVERSE 1.. NN
LOOP
FETCH C2 INTO  N1 ;

FETCH C3 INTO  MTH ;
 FETCH C1 INTO EENO,NAME1,NAME2;
EXIT WHEN C1%notfound;


if(N1=1) AND (MTH=8) then
CTN:=CTN+1;
DBMS_OUTPUT.PUT_LINE(EENO||'  '||NAME1||'  '||NAME2||''||I);
end if;
END LOOP;
DBMS_OUTPUT.PUT_LINE('NO. OF RECORD FOUND '||CTN);




CLOSE C1;
CLOSE C2;
CLOSE C3;
END;
/
OUTPUT:-
IN NORMAL  ORDER
NO. OF RECORD FOUND 0
IN REVERSE ORDER
NO. OF RECORD FOUND 0
```

```
====================================Query08================================
Write SQL code to create and execute an anonymous PL/SQL block that accept
staff ID from the console and will display staff details for said staff. A
system exception, NO_DATA_FOUND should be cached when the mentioned staff does
not exist.
===========================================================================

SQL> DECLARE
       S_id STAFF1.SID%type ;
       S_name STAFF1.NAME%type;
       BRCH STAFF1.BRANCH%type;
      DES STAFF1.DESG%TYPE;
      DT   STAFF1.JOIN_DT%TYPE;
    BEGIN
    S_ID:='&SID';
       SELECT  sid,name,branch,DESG,JOIN_DT INTO  S_id,S_name,
   BRCH,DES,DT FROM STAFF1
      WHERE SID = S_id;
         dbms_output.put_line('STAFF DETAIL IS AS FOLLOWS:-');
      DBMS_OUTPUT.PUT_LINE (S_id||' '||S_name||' ' ||BRCH||' '||DES||' '||DT
);


   EXCEPTION
      WHEN no_data_found THEN
         dbms_output.put_line('No such STAFF!');
      WHEN others THEN
         dbms_output.put_line('Error!');
   END;
   /
Enter value for sid: 102
old   8: S_ID:='&SID';
new   8: S_ID:='102';
STAFF DETAIL IS AS FOLLOWS:-
102 Adishesh Vidyarthi AIML Associate 22-JUL-06

PL/SQL procedure successfully completed.

SQL> /
Enter value for sid: 202
old   8: S_ID:='&SID';
new   8: S_ID:='202';
No such STAFF!

PL/SQL procedure successfully completed.


====================================Query09================================

Write SQL code to create and execute an anonymous PL/SQL block that defines
user-defined exceptions - BELOW_PAY_RANGE and ABOVE_PAY_RANGE. Your script
should accept an employee number from the console and check for the salary to
fall within the payscale [minpay, maxpay].
===========================================================================
SQL> CREATE TABLE PAYSCALE(DESIGNATION VARCHAR2(15) PRIMARY KEY, MINPAY
NUMBER(5),MAXPAY NUMBER(5),
   CHECK (DESIGNATION IN('Professor','Research Asst.','Asso. Professor',
    'Teaching Asst.', 'Asst. Professor')));
```

```
Table created.


SQL> INSERT INTO PAYSCALE VALUES('Asst. Professor',50000,90000);

1 row created.

SQL> INSERT INTO PAYSCALE VALUES('Teaching Asst.',20000,32500);

1 row created.

SQL>  INSERT INTO PAYSCALE VALUES('Research Asst.',30000,45000);
1 row created.

SQL> commit;
Commit complete.

SQL> select * from payscale;

DESIGNATION         MINPAY     MAXPAY
--------------- ---------- ----------
Asst. Professor      50000      90000
Teaching Asst.       20000      32500
Research Asst.       30000      45000

SQL> ALTER TABLE PAYSCALE MODIFY   MINPAY NUMBER(6);

Table altered.

SQL> ALTER TABLE PAYSCALE MODIFY   MAXPAY NUMBER(6);

Table altered.
SQL> INSERT INTO PAYSCALE VALUES ('Professor',140000,200000);

1 row created.
SQL> INSERT INTO PAYSCALE VALUES ('Asso. Professor',100000,140000);
1 row created.

SQL> COMMIT;
Commit complete.

SQL> SELECT * FROM PAYSCALE;

DESIGNATION         MINPAY     MAXPAY
--------------- ---------- ----------
Asst. Professor      50000      90000
Teaching Asst.       20000      32500
Research Asst.       30000      45000
Professor           140000     200000
Asso. Professor     100000     140000
```

```
DECLARE
ABOVE_PAY_RANGE  EXCEPTION;
```

```
BELOW_PAY_RANGE EXCEPTION;
CURSOR C1 (EENO NUMBER )IS SELECT MINPAY,MAXPAY,SALARY,ENO FROM PAYSCALE P
INNER JOIN EMPLOYEE E ON P.DESIGNATION=E.DESIGNATION WHERE ENO=EENO ;

SAL EMPLOYEE.SALARY%TYPE;
E_NO EMPLOYEE.ENO%TYPE;
MIN1 PAYSCALE.MINPAY%TYPE;
MAX1 PAYSCALE.MAXPAY%TYPE;
VR C1%ROWTYPE;

BEGIN
E_NO:='&E_NO';

OPEN C1 (E_NO);

FETCH C1 INTO MIN1,MAX1,SAL,E_NO;
IF(SAL<MIN1)THEN
RAISE BELOW_PAY_RANGE;
ELSIF(SAL>MAX1)THEN
RAISE ABOVE_PAY_RANGE ;

ELSIF(SAL >=MIN1)AND (SAL<=MAX1) THEN

dbms_output.put_line(E_NO||' RECIVES SALARY  '||SAL|| ' IN RANGE where MIN
='||MAX1||' AND MAX='||MIN1);
ELSE
RAISE no_data_found;

END IF;
CLOSE C1;
EXCEPTION
WHEN no_data_found THEN
        dbms_output.put_line('***** !DATA_FOUND! *****');
WHEN ABOVE_PAY_RANGE THEN

        dbms_output.put_line('EXCEPTION');

        dbms_output.put_line(E_NO||' RECIVES SALARY  '||SAL|| ' ABOVE SCALE
WHERE MAX_PAYSCALE IS '||MAX1||' MIN_PAYSCALE  '||MIN1);
WHEN BELOW_PAY_RANGE THEN
        dbms_output.put_line('EXCEPTION');

        dbms_output.put_line(E_NO||' RECIVES SALARY '||SAL||' BELOW SCALE
MAX_PAYSCALE IS '||MAX1||' AND MIN_PAYSCALE IS  '||MIN1);


END;
/
Output:-
Enter value for e_no: 20
old  13: E_NO:='&E_NO';
new  13: E_NO:='20';
***** !DATA_FOUND! *****


Enter value for e_no: 20
old  13: E_NO:='&E_NO';
new  13: E_NO:='20';
***** !DATA_FOUND! *****
```

```
PL/SQL procedure successfully completed.

SQL> /
Enter value for e_no: 7101
old  13: E_NO:='&E_NO';
new  13: E_NO:='7101';
7101 RECIVES SALARY  150000 IN RANGE where MIN =200000 AND MAX=140000

PL/SQL procedure successfully completed.

SQL> /
Enter value for e_no: 7104
old  13: E_NO:='&E_NO';
new  13: E_NO:='7104';
EXCEPTION
7104 RECIVES SALARY 138400 BELOW SCALE MAX_PAYSCALE IS 200000 AND MIN_PAYSCALE
IS  140000

PL/SQL procedure successfully completed.

SQL> /
Enter value for e_no: 7106
old  13: E_NO:='&E_NO';
new  13: E_NO:='7106';
7106 RECIVES SALARY  127400 IN RANGE where MIN =140000 AND MAX=100000

PL/SQL procedure successfully completed.

SQL> /
Enter value for e_no: 7109
old  13: E_NO:='&E_NO';
new  13: E_NO:='7109';
EXCEPTION
7109 RECIVES SALARY  91000 ABOVE SCALE  WHERE MAX_PAYSCALE IS 90000
MIN_PAYSCALE
50000

PL/SQL procedure successfully completed.

SQL> /
Enter value for e_no: 7111
old  13: E_NO:='&E_NO';
new  13: E_NO:='7111';
EXCEPTION
7111 RECIVES SALARY  48200 ABOVE SCALE  WHERE MAX_PAYSCALE IS 45000
MIN_PAYSCALE
30000

PL/SQL procedure successfully completed.

SQL> /
Enter value for e_no: 7114
old  13: E_NO:='&E_NO';
new  13: E_NO:='7114';
EXCEPTION
7114 RECIVES SALARY  32850 ABOVE SCALE  WHERE MAX_PAYSCALE IS 32500
MIN_PAYSCALE
20000
```

```
PL/SQL procedure successfully completed.

SQL> /
Enter value for e_no: 7117
old  13: E_NO:='&E_NO';
new  13: E_NO:='7117';
7117 RECIVES SALARY  32200 IN RANGE where MIN =32500 AND MAX=20000

PL/SQL procedure successfully completed.
```

==========Query10==========

Write a SQL code to create and execute an anonymous PL/SQL block that will modify Query-09 to process all records of EMPLOYEE table. You need not acquire employee number from console. You should only report the violations.

==========

```
SQL> DECLARE
   ABOVE_PAY_RANGE  EXCEPTION;
    BELOW_PAY_RANGE EXCEPTION;
    E_NO EMPLOYEE.ENO%TYPE;
    ALL_REC NUMBER(5);
    MIN1 PAYSCALE.MINPAY%TYPE;
    MAX1 PAYSCALE.MAXPAY%TYPE;
    SAL EMPLOYEE.SALARY%TYPE;

   CURSOR C1 (EENO NUMBER )IS SELECT MINPAY,MAXPAY,SALARY,ENO FROM PAYSCALE P
INNER JOIN EMPLOYEE E ON P.DESIGNATION=E.DESIGNATION WHERE ENO=EENO ;

   BEGIN
   SELECT COUNT(* ) INTO ALL_REC FROM EMPLOYEE;
   FOR I IN 1.. ALL_REC
   LOOP
   SELECT ENO INTO E_NO FROM EMPLOYEE WHERE ENO=I+7100;
   OPEN C1(E_NO);
   FETCH C1  INTO MIN1,MAX1,SAL,E_NO;
   EXIT WHEN C1%NOTFOUND;
     --dbms_output.put_line(E_NO);
   IF(SAL>MAX1) THEN
    dbms_output.put_line(E_NO||' '||'ABOVE_PAY_RANGE')  ;
   END IF;
   IF(SAL<MIN1) THEN
       dbms_output.put_line(E_NO||' '||'BELOW_PAY_RANGE')  ;
   END IF;
   IF(SAL>=MIN1)AND (SAL<=MAX1) THEN
       dbms_output.put_line(E_NO||' HAVING SALARY'||SAL||'MIN SAL'||MIN1||'
'||MAX1);
   END IF;
   CLOSE C1;
   END LOOP;
   EXCEPTION
   when NO_DATA_FOUND then
    DBMS_OUTPUT.PUT_LINE('Caught raised exception NO_DATA_FOUND');
   WHEN ABOVE_PAY_RANGE THEN
           dbms_output.put_line(E_NO||' RECIVES SALARY  '||SAL|| ' ABOVE SCALE
'||MAX1||'  '||MIN1);
   WHEN BELOW_PAY_RANGE THEN
           dbms_output.put_line(E_NO||' RECIVES SALARY '||SAL||' BELOW SCALE
'||MAX1||'  '||MIN1);
```

```
   WHEN OTHERS THEN
   dbms_output.put_line('exceptiON');
   END;
   /
7101 HAVING SALARY150000MIN SAL140000 200000
7102 HAVING SALARY146500MIN SAL140000 200000
7103 HAVING SALARY148000MIN SAL140000 200000
7104 BELOW_PAY_RANGE
7105 HAVING SALARY127400MIN SAL100000 140000
7106 HAVING SALARY127400MIN SAL100000 140000
7107 HAVING SALARY127400MIN SAL100000 140000
7108 HAVING SALARY119700MIN SAL100000 140000
7109 ABOVE_PAY_RANGE
7110 HAVING SALARY86400MIN SAL50000 90000
7111 ABOVE_PAY_RANGE
7112 HAVING SALARY44600MIN SAL30000 45000
7113 ABOVE_PAY_RANGE
7114 ABOVE_PAY_RANGE
7115 HAVING SALARY30000MIN SAL20000 32500
7116 HAVING SALARY30000MIN SAL20000 32500
7117 HAVING SALARY32200MIN SAL20000 32500

PL/SQL procedure successfully completed.

[PART-2]
SQL> DECLARE
   ABOVE_PAY_RANGE  EXCEPTION;
    BELOW_PAY_RANGE EXCEPTION;
    E_NO EMPLOYEE.ENO%TYPE;
    ALL_REC NUMBER(5);
    MIN1 PAYSCALE.MINPAY%TYPE;
    MAX1 PAYSCALE.MAXPAY%TYPE;
    SAL EMPLOYEE.SALARY%TYPE;

   CURSOR C1 (EENO NUMBER )IS SELECT MINPAY,MAXPAY,SALARY,ENO FROM PAYSCALE P
INNER JOIN EMPLOYEE E ON P.DESIGNATION=E.DESIGNATION WHERE ENO=EENO ;

   BEGIN
   SELECT COUNT(* ) INTO ALL_REC FROM EMPLOYEE;
   FOR I IN 1.. ALL_REC
   LOOP
   SELECT ENO INTO E_NO FROM EMPLOYEE WHERE ENO=I+7100;
   OPEN C1(E_NO);
   FETCH C1  INTO MIN1,MAX1,SAL,E_NO;
   EXIT WHEN C1%NOTFOUND;
     --dbms_output.put_line(E_NO);
   IF(SAL>MAX1) THEN
    dbms_output.put_line(E_NO||' '||'ABOVE_PAY_RANGE')  ;
   END IF;
   IF(SAL<MIN1) THEN
       dbms_output.put_line(E_NO||' '||'BELOW_PAY_RANGE')  ;
   END IF;
   CLOSE C1;
   END LOOP;
   EXCEPTION
   when NO_DATA_FOUND then
    DBMS_OUTPUT.PUT_LINE('Caught raised exception NO_DATA_FOUND');
   WHEN ABOVE_PAY_RANGE THEN
```

```
           dbms_output.put_line(E_NO||' RECIVES SALARY  '||SAL|| ' ABOVE SCALE
'||MAX1||'  '||MIN1);
   WHEN BELOW_PAY_RANGE THEN
           dbms_output.put_line(E_NO||' RECIVES SALARY '||SAL||' BELOW SCALE
'||MAX1||'  '||MIN1);
   WHEN OTHERS THEN
   dbms_output.put_line('exceptiON');
   END;
   /
OUTPUT:-
7104 BELOW_PAY_RANGE
7109 ABOVE_PAY_RANGE
7111 ABOVE_PAY_RANGE
7113 ABOVE_PAY_RANGE
7114 ABOVE_PAY_RANGE
*********************************************************************************
```

Viva Voice
==========

**********Question-1**********
What is an anonymous block?
Ans:-
An anonymous block is an **unnamed sequence of actions**. Since they are unnamed, anonymous blocks cannot be referenced by other program units. In contrast to anonymous blocks, stored/ named code blocks include Packages, Procedures, and Functions . Here are some example anonymous blocks written in PL/SQL :

**********Question-2**********
What is an exception? List the standard PL/SQL exceptions.
Ans:-
An exception is an error condition during a program execution. PL/SQL supports programmers to catch such conditions using **EXCEPTION** block in the program and an appropriate action is taken against the error condition. There are two types of exceptions –

- System-defined exceptions
- User-defined exceptions

LIST:-

1. ACCESS_INTO_NULL
2. CASE_NOT_FOUND
3. COLLECTION_IS_NULL
4. PROGRAM_ERROR
5. ZERO_DIVIDE

**********Question-3**********
Differentiate between a relation and table.

Ans:
1.& - Bitwise AND
2.&&- logical And

**********Question-4**********
4.Why it is a good practice to use %TYPE when declaring variables?
ANS:-As per me %type is use when we need to use only specific and limited data elements from the table.or else I will use rowtype%;

****

Inference:-
==========

1. Cursor is use to retrive multiple set of data.
2. %type is use when we don’t know about data type of column.
3. The %ROWTYPE attribute is used to **define a record with fields corresponding to all of the columns that are fetched from a cursor or cursor variable**