```
 --JOURNAL COMPILATION
 /*
        AIM:To write and execute PL/SQL blocks (with exception handling)
            including PL/SQL subprograms using Oracle 11g.

        PROBLEM STATEMENT:Use EMPLOYEE table established in Experiment-01 to
                          create and execute the anonymous PL/SQL block queries.
                          The schema for the EMPLOYEE table is -
                             EMPLOYEE (EID, FNAME, LNAME, BIRTHDATE, GENDER, SSN,
                             HIREDATE, SALARY, DEPARTMENT, DESIGNATION)
 */

 ------------------------------------ QUERY-01 ------------------------------
 /*
 Write a SQL code to write and execute an anonymous PL/SQL block that will insert 10
tuples into TEST_TBL.
 Ensure to commit the populated records. Test the insertion in TEST_TBL by displaying
its contents.
 */
 ----------------------------------------------------------------------------

SQL> CREATE TABLE TEST_TBL(
  2  REC_NO NUMBER(3),
  3  CURR_DT DATE,
  4  CONSTRAINT TEST_TBL_PK PRIMARY KEY (REC_NO),
  5  CONSTRAINT TEST_TBL_CHK_RANGE CHECK (REC_NO BETWEEN 101 AND 999)
  6  );

Table created.

SQL> DECLARE
  2          BASE_CNT CONSTANT INT :=100;
  3          CNT INT;
  4  BEGIN
  5          FOR CNT IN 1 .. 10 LOOP
  6                  INSERT INTO TEST_TBL(REC_NO,CURR_DT)
  7                          VALUES(BASE_CNT+CNT,SYSDATE);
  8          END LOOP;
  9      COMMIT;
 10  END;
 11  /

PL/SQL procedure successfully completed.

SQL> SELECT * FROM TEST_TBL;

    REC_NO CURR_DT

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

       101 11-APR-17

       102 11-APR-17

       103 11-APR-17

       104 11-APR-17

       105 11-APR-17
       106 11-APR-17

       107 11-APR-17

       108 11-APR-17

       109 11-APR-17

       110 11-APR-17


10 rows selected.

SQL> CREATE TABLE EMPP AS SELECT EID,FNAME||' '||LNAME AS ENAME,HIREDATE,SALARY FROM
EMPLOYEE WHERE 1=2;

Table created.

SQL> DESC EMPP;
 Name                      Null?    Type
 ------------------------- -------- ---------------
 EID                       NOT NULL NUMBER(4)
 ENAME                              VARCHAR2(21)
 HIREDATE                  NOT NULL DATE
 SALARY                    NOT NULL NUMBER(7,2)

SQL> SELECT COUNT(*) FROM EMPP;

  COUNT(*)

----------

         0


1 row selected.

SQL> SELECT CONSTRAINT_NAME FROM USER_CONSTRAINTS WHERE TABLE_NAME LIKE 'EMPP';

CONSTRAINT_NAME

---------------------
SYS_C0012395

SYS_C0012394

SYS_C0012393


3 rows selected.

 ------------------------------------ QUERY-02 ------------------------------
 /*
 Write a SQL code to write and execute an anonymous PL/SQL block that will use %TYPE
variables to populate
 the EMPP table with corresponding tuples in EMPLOYEE table.
 */
 ---------------------------------------------------------------------------
SQL> DECLARE
  2  EMPID EMPLOYEE.EID%TYPE;
  3  ENAME VARCHAR2(40);
  4  HDATE EMPLOYEE.HIREDATE%TYPE;
  5  SAL EMPLOYEE.SALARY%TYPE;
  6  CNT INT;
  7  LCNT INT;
  8  BASE_CNT CONSTANT INT :=7100;
  9  BEGIN
 10  SELECT COUNT(*) INTO CNT FROM EMPLOYEE;
 11  FOR LCNT IN 1..CNT LOOP
 12      SELECT EID,(FNAME||' '||LNAME),HIREDATE,SALARY INTO
 13      EMPID,ENAME,HDATE,SAL FROM EMPLOYEE
 14      WHERE EID=BASE_CNT+LCNT;
 15      INSERT INTO EMPP VALUES(EMPID,ENAME,HDATE,SAL);
 16  END LOOP;
 17  END;
 18  /

PL/SQL procedure successfully completed.

SQL> SELECT * FROM EMPP;

       EID ENAME                HIREDATE      SALARY

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

      7101 SAMANTHA JONES       08-NOV-94      16500

      7102 ALBERT GREENFIELD    12-JUL-98      14200

      7103 JULIA MARIN          01-DEC-99      13320

      7104 MARTINA JACOBSON     15-NOV-96      15550

      7105 ALEXANDER LLOYD      01-FEB-94      17500

      7106 WILLIAM SMITHFIELD   23-JUN-96      15660

      7107 EUGENE SABATINI      10-OCT-94      16500

      7108 JAMES WASHNGTON      22-AUG-98      14000

      7109 LARRY GOMES          18-MAY-99      13650


9 rows selected.

SQL> ROLLBACK;

Rollback complete.

------------------------------------ QUERY-03 ------------------------------
 /*
 Write a SQL code to write and execute an anonymous PL/SQL block that will use %ROWTYPE
variables
 to populate the EMPP table with corresponding tuples in EMPLOYEE table.
 */
 ----------------------------------------------------------------------------
SQL> DECLARE
  2  EMP_REC EMPLOYEE%ROWTYPE;
  3  CNT INT;
  4  LCNT INT;
  5  BASE_CNT CONSTANT INT :=7100;
  6  BEGIN
  7  SELECT COUNT(*) INTO CNT FROM EMPLOYEE;
  8  FOR LCNT IN 1..CNT LOOP
  9    SELECT * INTO EMP_REC FROM EMPLOYEE WHERE EID=BASE_CNT+LCNT;
 10    INSERT INTO EMPP VALUES(EMP_REC.EID,EMP_REC.FNAME||'
'||EMP_REC.LNAME,EMP_REC.HIREDATE,EMP_REC.SALARY);
 11  END LOOP;
 12  END;
 13  /

PL/SQL procedure successfully completed.

SQL> SELECT * FROM EMPP;

       EID ENAME                HIREDATE      SALARY

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

      7101 SAMANTHA JONES       08-NOV-94      16500

      7102 ALBERT GREENFIELD    12-JUL-98      14200

      7103 JULIA MARIN          01-DEC-99      13320

      7104 MARTINA JACOBSON     15-NOV-96      15550

      7105 ALEXANDER LLOYD      01-FEB-94      17500

      7106 WILLIAM SMITHFIELD   23-JUN-96      15660

      7107 EUGENE SABATINI      10-OCT-94      16500

      7108 JAMES WASHNGTON      22-AUG-98      14000

      7109 LARRY GOMES          18-MAY-99      13650


9 rows selected.

 ----------------------------------- QUERY-04 ------------------------------
 /*
 Write a SQL code to write and execute an anonymous PL/SQL block that will display the
contents
 of EMPP table without using declared variables. You should format the output using
RPAD() and/or LPAD(),
 while including proper headers in the result.
 */
 ---------------------------------------------------------------------------
SQL> --VAL IS BY DEFAULT ASSUMING A ROWTYPE
SQL> BEGIN
  2  DBMS_OUTPUT.PUT_LINE('EID      ENAME               HIREDATE      SALARY ');
  3  FOR VAL IN(SELECT EID,ENAME,HIREDATE,SALARY FROM EMPP) LOOP
  4    DBMS_OUTPUT.PUT_LINE(RPAD(VAL.EID,8,' ')||' '||RPAD(VAL.ENAME,20,' ')||'
'||RPAD(VAL.HIREDATE,12,' ')||' '||LPAD(VAL.SALARY,6,' '));
  5  END LOOP;
  6  END;
  7  /
EID      ENAME                HIREDATE      SALARY
```

```
7101     SAMANTHA JONES       08-NOV-94     16500

7102     ALBERT GREENFIELD    12-JUL-98     14200

7103     JULIA MARIN          01-DEC-99     13320

7104     MARTINA JACOBSON     15-NOV-96     15550

7105     ALEXANDER LLOYD      01-FEB-94     17500

7106     WILLIAM SMITHFIELD   23-JUN-96     15660

7107     EUGENE SABATINI      10-OCT-94     16500

7108     JAMES WASHNGTON      22-AUG-98     14000

7109     LARRY GOMES          18-MAY-99     13650


PL/SQL procedure successfully completed.

 ----------------------------------- QUERY-05 ------------------------------
 /*
 Write a SQL query to find the Oracle Database version and the PL/SQL version running
currently
 on your machine. Use V$VERSION view of Oracle.Write a SQL code to write and execute an
anonymous
 PL/SQL block that will display the current time-stamp of the system. Also display the
time-stamp 3 hours before.
 */
 ---------------------------------------------------------------------------
SQL> SELECT BANNER FROM V$VERSION;

BANNER

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

Oracle Database 12c Enterprise Edition Release 12.1.0.2.0 - 64bit Production

PL/SQL Release 12.1.0.2.0 - Production

CORE    12.1.0.2.0      Production

TNS for 64-bit Windows: Version 12.1.0.2.0 - Production

NLSRTL Version 12.1.0.2.0 - Production


5 rows selected.

SQL> BEGIN
  2  DBMS_OUTPUT.PUT_LINE(SYSTIMESTAMP);
  3  DBMS_OUTPUT.PUT_LINE(SYSTIMESTAMP- interval '3' hour);
  4  END;
  5  /
11-APR-17 12.08.16.036000000 AM +05:30

10-APR-17 09.08.16.036000000 PM +05:30
```

```
PL/SQL procedure successfully completed.

  ----------------------------------- QUERY-06 ------------------------------
 /*
 Write a SQL code to write and execute an anonymous PL/SQL block that will display the
system date.
 Use exception (exception TVALUE_ERROR) to check if the variable holding the system date
is large enough in size.
 Re-execute the block with appropriate modification to test the exception.
 */
 ---------------------------------------------------------------------------
SQL> DECLARE
  2  D VARCHAR2(5);
  3  BEGIN
  4  D:=TO_CHAR(SYSDATE);
  5  DBMS_OUTPUT.PUT_LINE(D);
  6  EXCEPTION
  7  WHEN VALUE_ERROR THEN
  8  DBMS_OUTPUT.PUT_LINE('VALUE ERROR OCCURED');
  9  END;
 10  /
VALUE ERROR OCCURED


PL/SQL procedure successfully completed.

SQL> DECLARE
  2  D VARCHAR2(50);
  3  BEGIN
  4  D:=TO_CHAR(SYSDATE);
  5  DBMS_OUTPUT.PUT_LINE(D);
  6  EXCEPTION
  7  WHEN VALUE_ERROR THEN
  8  DBMS_OUTPUT.PUT_LINE(' VALUE ERROR OCCURED');
  9  END;
 10  /
11-APR-17


PL/SQL procedure successfully completed.

 ----------------------------------- QUERY-07 ------------------------------
 /*
 Write a SQL code to write and execute an anonymous PL/SQL block that will check (say,
for employee number 7103)
 whether an employee is entitled to receive the longevity bonus. Longevity bonus is
given to employees who has
 been with the company for at least 20 years.Now, re-execute the block to extend
longevity bonus to employees
 with 15 years of service.
 */
 ---------------------------------------------------------------------------
SQL> DECLARE
  2  EMPREC EMPLOYEE%ROWTYPE;
  3  BASE_CNT CONSTANT INT :=7100;
  4  CYEAR NUMBER;
  5  HYEAR NUMBER;
  6  YEARDIFF NUMBER;
  7  CNT INT;
  8  LCNT INT;
  9  YEARCNT INT :=20;
 10  BEGIN
 11  SELECT EXTRACT(YEAR FROM SYSDATE) INTO CYEAR FROM DUAL;
 12  SELECT COUNT(*) INTO CNT FROM EMPLOYEE;
 13  <<TWICELOOPER>>
 14  DBMS_OUTPUT.PUT_LINE('EMPLOYEE WORKING WITH COMPANY FOR ATLEAST '||YEARCNT||'
YEARS');
 15  FOR LCNT IN 1 .. CNT LOOP
 16      SELECT * INTO EMPREC FROM EMPLOYEE WHERE EID=BASE_CNT+LCNT;
 17      SELECT EXTRACT(YEAR FROM EMPREC.HIREDATE) INTO HYEAR FROM DUAL;
 18      YEARDIFF:=CYEAR-HYEAR;
 19      IF YEARDIFF >=YEARCNT THEN
 20         DBMS_OUTPUT.PUT_LINE(EMPREC.EID||' '||EMPREC.FNAME||' '||EMPREC.LNAME);
 21      END IF;
 22  END LOOP;
 23  IF YEARCNT=20 THEN
 24      YEARCNT:=15;
 25      GOTO TWICELOOPER;
 26  END IF;
 27  END;
 28  /
EMPLOYEE WORKING WITH COMPANY FOR ATLEAST 20 YEARS

7101 SAMANTHA JONES

7104 MARTINA JACOBSON

7105 ALEXANDER LLOYD

7106 WILLIAM SMITHFIELD

7107 EUGENE SABATINI

EMPLOYEE WORKING WITH COMPANY FOR ATLEAST 15 YEARS

7101 SAMANTHA JONES

7102 ALBERT GREENFIELD

7103 JULIA MARIN

7104 MARTINA JACOBSON

7105 ALEXANDER LLOYD

7106 WILLIAM SMITHFIELD

7107 EUGENE SABATINI

7108 JAMES WASHNGTON

7109 LARRY GOMES


PL/SQL procedure successfully completed.

 ------------------------------------ QUERY-08 ------------------------------
 /*
  Write a SQL code to write and execute an anonymous PL/SQL block that will locate
  the first November-born employee.
 */
 ---------------------------------------------------------------------------
SQL> DECLARE
  2  EMP_REC EMPLOYEE%ROWTYPE;
  3  CNT INT;
  4  LCNT INT;
  5  MNTH INT;
  6  BASE_CNT CONSTANT INT :=7100;
  7  BEGIN
  8  SELECT COUNT(*) INTO CNT FROM EMPLOYEE;
  9  FOR LCNT IN 1..CNT LOOP
 10    SELECT * INTO EMP_REC FROM EMPLOYEE WHERE EID=BASE_CNT+LCNT;
 11    SELECT EXTRACT(MONTH FROM EMP_REC.BIRTHDATE) INTO MNTH FROM DUAL;
 12    IF MNTH=11 THEN
 13        DBMS_OUTPUT.PUT_LINE(EMP_REC.FNAME||' '||EMP_REC.LNAME|| '
'||EMP_REC.BIRTHDATE);
 14        EXIT;
 15    END IF;
 16  END LOOP;
 17  END;
 18  /
WILLIAM SMITHFIELD 02-NOV-72


PL/SQL procedure successfully completed.

 ----------------------------------- QUERY-09 ------------------------------
 /*
 Write a SQL code to write and execute an anonymous PL/SQL block that will locate the
first
 November-born employee, when EMPLOYEE table is searched in reversed order.
 */
 ---------------------------------------------------------------------------
SQL> DECLARE
  2  EMP_REC EMPLOYEE%ROWTYPE;
  3  CNT INT;
  4  LCNT INT;
  5  MNTH INT;
  6  BASE_CNT CONSTANT INT :=7100;
  7  BEGIN
  8  SELECT COUNT(*) INTO CNT FROM EMPLOYEE;
  9  FOR LCNT IN REVERSE 1..CNT LOOP
 10    SELECT * INTO EMP_REC FROM EMPLOYEE WHERE EID=BASE_CNT+LCNT;
 11    SELECT EXTRACT(MONTH FROM EMP_REC.BIRTHDATE) INTO MNTH FROM DUAL;
 12    IF MNTH=11 THEN
 13        DBMS_OUTPUT.PUT_LINE(EMP_REC.FNAME||' '||EMP_REC.LNAME|| '
'||EMP_REC.BIRTHDATE);
 14        EXIT;
 15    END IF;
 16  END LOOP;
 17  END;
 18  /
EUGENE SABATINI 09-NOV-73


PL/SQL procedure successfully completed.

 ----------------------------------- QUERY-10 ------------------------------
 /*
 Write a SQL code to write and execute an anonymous PL/SQL block that accept an employee
number
 from the console and will display employee information for said employee
 (minimal output -- Employee Number, Name of Employee, Designation, Salary).
 A system exception, NO_DATA_FOUND should be cached when the mentioned employee does not
exist.
 */
 -----------------------------------------------------------------------------
SQL> DECLARE
  2  EMPID EMPLOYEE.EID%TYPE;
  3  EREC EMPLOYEE%ROWTYPE;
  4  BEGIN
  5  DBMS_OUTPUT.PUT_LINE('ENTER EMPLOYEE ID:');
  6  EMPID:='&EMPID';
  7  SELECT * INTO EREC FROM EMPLOYEE WHERE EID=EMPID;
  8  DBMS_OUTPUT.PUT_LINE(EREC.EID||'  '||EREC.LNAME||'  '||EREC.DESIGNATION||'
'||EREC.SALARY);
  9  EXCEPTION
 10    WHEN NO_DATA_FOUND THEN
 11      DBMS_OUTPUT.PUT_LINE('NO RECORD EXIST WITH EID:='||EMPID);
 12  END;
 13  /
Enter value for empid: 7101
old   6: EMPID:='&EMPID';
new   6: EMPID:='7101';
ENTER EMPLOYEE ID:

7101  JONES  PROFESSOR 16500


PL/SQL procedure successfully completed.

 ------------------------------------ QUERY-11 -----------------------------
 /*
 Write a SQL code to write and execute an anonymous PL/SQL block that defines
 user-defined exceptions - BELOW_PAY_RANGE and ABOVE_PAY_RANGE.
 */
 -----------------------------------------------------------------------------
SQL> DROP TABLE PAYSCALE;

Table dropped.

SQL> CREATE TABLE PAYSCALE(
  2  DESIGNATION VARCHAR(15),
  3  MINPAY NUMBER(5),
  4  MAXPAY NUMBER(5),
  5  CONSTRAINT PAYSCALE_PK PRIMARY KEY (DESIGNATION),
  6  CONSTRAINT PAYSCALE_CHK_DESIGNATION CHECK (DESIGNATION IN('PROFESSOR','SR.
LECTURER','LECTURER','ASST. PROFESSOR'))
  7  );

Table created.

SQL> INSERT INTO PAYSCALE VALUES('LECTURER', 12000, 13500);
1 row created.
SQL> INSERT INTO PAYSCALE VALUES('SR. LECTURER', 13000, 15000);
1 row created.
SQL> INSERT INTO PAYSCALE VALUES('ASST. PROFESSOR', 14500, 16500);
1 row created.
SQL> INSERT INTO PAYSCALE VALUES('PROFESSOR', 16000, 19000);
```

```
1 row created.

SQL> COMMIT;

Commit complete.

SQL> DECLARE
  2  BELOW_PAY_RANGE EXCEPTION;
  3  ABOVE_PAY_RANGE EXCEPTION;
  4  EMPID EMPLOYEE.EID%TYPE;
  5  EXPNO EMPLOYEE.EID%TYPE;
  6  EREC EMPLOYEE%ROWTYPE;
  7  PAYSCALEREC PAYSCALE%ROWTYPE;
  8  EXPMINPAY PAYSCALE.MINPAY%TYPE;
  9  EXPMAXPAY PAYSCALE.MAXPAY%TYPE;
 10  BEGIN
 11  DBMS_OUTPUT.PUT_LINE('ENTER EID OF THE EMPLOYEE:');
 12  EMPID:='&EMPID';
 13  SELECT * INTO EREC FROM EMPLOYEE WHERE EID=EMPID;
 14  SELECT * INTO PAYSCALEREC FROM PAYSCALE WHERE DESIGNATION=EREC.DESIGNATION;
 15        EXPNO:=EREC.EID;
 16        EXPMINPAY:=PAYSCALEREC.MINPAY;
 17        EXPMAXPAY:=PAYSCALEREC.MAXPAY;
 18  IF EREC.SALARY >PAYSCALEREC.MINPAY THEN
 19      IF EREC.SALARY < PAYSCALEREC.MAXPAY THEN
 20        DBMS_OUTPUT.PUT_LINE(EREC.EID||' RECEIVES SALARY IN SCALE
['||PAYSCALEREC.MINPAY||','||PAYSCALEREC.MAXPAY||']');
 21      ELSE
 22        RAISE ABOVE_PAY_RANGE;
 23      END IF;
 24  ELSE
 25      RAISE BELOW_PAY_RANGE;
 26  END IF;
 27  EXCEPTION
 28    WHEN BELOW_PAY_RANGE THEN
 29        DBMS_OUTPUT.PUT_LINE(EXPNO||' Receives Salary Below Scale
['||EXPMINPAY||','||EXPMAXPAY||']');
 30    WHEN ABOVE_PAY_RANGE THEN
 31        DBMS_OUTPUT.PUT_LINE(EXPNO||' Receives Salary Above Scale
['||EXPMINPAY||','||EXPMAXPAY||']');
 32    WHEN NO_DATA_FOUND THEN
 33        DBMS_OUTPUT.PUT_LINE('NO RECORDS FOUND WITH EID:='||EXPNO);
 34    WHEN OTHERS THEN
 35        DBMS_OUTPUT.PUT_LINE('SOMETHING NOT
CORRECT'||TO_CHAR(SQLCODE)||'::'||TO_CHAR(SQLERRM));
 36  END;
 37  /
Enter value for empid: 7106
old  12: EMPID:='&EMPID';
new  12: EMPID:='7106';
ENTER EID OF THE EMPLOYEE:

7106 RECEIVES SALARY IN SCALE [14500,16500]


PL/SQL procedure successfully completed.

 ---------------------------------- QUERY-12 ------------------------------
 /*
 Write a SQL code to write and execute an anonymous PL/SQL block that will modify
```

```
Query-11** to process
 all records of EMPLOYEE table. You need not acquire employee number from console.
 You should only report the violations.
 */
 ----------------------------------------------------------------------------
SQL> DECLARE
  2  EMPID EMPLOYEE.EID%TYPE;
  3  EXPNO EMPLOYEE.EID%TYPE;
  4  EREC EMPLOYEE%ROWTYPE;
  5  PAYSCALEREC PAYSCALE%ROWTYPE;
  6  EXPMINPAY PAYSCALE.MINPAY%TYPE;
  7  EXPMAXPAY PAYSCALE.MAXPAY%TYPE;
  8  CNT INT;
  9  LCNT INT;
 10  BASE_CNT CONSTANT INT :=7100;
 11  BEGIN
 12  SELECT COUNT(*) INTO CNT FROM EMPLOYEE;
 13  FOR LCNT IN 1..CNT LOOP
 14  EMPID:=BASE_CNT+LCNT;
 15  SELECT * INTO EREC FROM EMPLOYEE WHERE EID=EMPID;
 16  SELECT * INTO PAYSCALEREC FROM PAYSCALE WHERE DESIGNATION=EREC.DESIGNATION;
 17        EXPNO:=EREC.EID;
 18        EXPMINPAY:=PAYSCALEREC.MINPAY;
 19        EXPMAXPAY:=PAYSCALEREC.MAXPAY;
 20  IF EREC.SALARY >PAYSCALEREC.MINPAY THEN
 21      IF EREC.SALARY < PAYSCALEREC.MAXPAY THEN
 22        DBMS_OUTPUT.PUT_LINE(EREC.EID||' RECEIVES SALARY IN SCALE
['||PAYSCALEREC.MINPAY||','||PAYSCALEREC.MAXPAY||']');
 23      ELSE
 24        DBMS_OUTPUT.PUT_LINE(EXPNO||' Receives Salary Above Scale
['||EXPMINPAY||','||EXPMAXPAY||']');
 25      END IF;
 26  ELSE
 27      DBMS_OUTPUT.PUT_LINE(EXPNO||' Receives Salary Below Scale
['||EXPMINPAY||','||EXPMAXPAY||']');
 28  END IF;
 29  END LOOP;
 30  EXCEPTION
 31    WHEN NO_DATA_FOUND THEN
 32        DBMS_OUTPUT.PUT_LINE('NO RECORDS FOUND WITH EID:='||EXPNO);
 33    WHEN OTHERS THEN
 34        DBMS_OUTPUT.PUT_LINE('SOMETHING NOT
CORRECT'||TO_CHAR(SQLCODE)||'::'||TO_CHAR(SQLERRM));
 35  END;
 36  /
7101 RECEIVES SALARY IN SCALE [16000,19000]

7102 RECEIVES SALARY IN SCALE [13000,15000]

7103 RECEIVES SALARY IN SCALE [12000,13500]

7104 RECEIVES SALARY IN SCALE [14500,16500]

7105 RECEIVES SALARY IN SCALE [16000,19000]

7106 RECEIVES SALARY IN SCALE [14500,16500]

7107 RECEIVES SALARY IN SCALE [16000,19000]

7108 RECEIVES SALARY IN SCALE [13000,15000]
```

```
7109 Receives Salary Above Scale [12000,13500]


PL/SQL procedure successfully completed.

 --------------------------------- END OF QUERIES---------------------------

SQL> SET FEEDBACK OFF
SQL> SPOOL OFF
```