Winter – SEMESTER 2022 - 23 **Name:** Nidhi Singh
Course Code: MCSE506P **Reg. No:**22MAI0015
Course-Title: – Database Systems Lab
DIGITAL ASSIGNMENT - V
(LAB)

Slot- L29+L30

Faculty: Dr. KARTHIK G M - SCOPE

EXCEPTION HANDLING IN PL/SQL

8. DISPLAY THE NAME OF THAT MANAGER WHO HAS JOINED THE ORGANIZATION IN THE YEAR 1999. IF THERE IS NO OUTPUT OR IF THERE IS MORE THAN ONE OUTPUT THEN INSTEAD OF GIVING ERROR THE PROGRAM MUST GIVE PROPER MESSAGE.

```
SQL> ed

DECLARE
        vename emp.ename % TYPE;
        vjob emp.job % TYPE;
        error_msg VARHCAR(100);
        error_code NUMBER;
BEGIN
        SELECT ename,job INTO vename,vjob FROM emp WHERE hiredate
                BETWEEN '01-jan-09' AND '31-dec-09';

        dbms_output.put_line(vename||' '||vjob);

        EXCEPTION
                WHEN no_data_found THEN
                        dbms_output.put_line('no employee hired in 09');
                WHEN too_many_rows THEN
                        dbms_output.put_line('more than one manager has joined in
                                09');
                WHEN others THEN
                        error_msg:=SUBSTR(sqlerrm,1,100);
                        error_code:=SQLCODE;
                        dbms_output.put_line('error during block execution');
                        dbms_output.put_line(error_msg||' '||error_code);
END;
/
```

```
SQL> ed
Wrote file afiedt.buf

  1   DECLARE
  2  vename emp.ename % TYPE;
  3  vjob emp.job % TYPE;
  4  error_msg VARCHAR(100);
  5  error_code NUMBER;
  6  BEGIN
  7  SELECT ename,job INTO vename,vjob FROM emp WHERE hiredate
  8  BETWEEN '08-sep-81' AND '23-jan-82';
  9  dbms_output.put_line(vename||' '||vjob);
 10  EXCEPTION
 11  WHEN no_data_found THEN
 12  dbms_output.put_line('no employee hired in 09');
 13  WHEN too_many_rows THEN
 14  dbms_output.put_line('more than one manager has joined in
 15  09');
 16  WHEN others THEN
 17  error_msg:=SUBSTR(sqlerrm,1,100);
 18  error_code:=SQLCODE;
 19  dbms_output.put_line('error during block execution');
 20  dbms_output.put_line(error_msg||' '||error_code);
 21* END;
SQL> /
more than one manager has joined in
09

PL/SQL procedure successfully completed.
```

SUBPROGRAMS IN PL/SQL USING PROCEDURES & FUNCTIONS

9A. CREATE A PROCEDURE INCR AND INCREASE THE SALARY OF EMPLOYEE.

```
CREATE or REPLACE PROCEDURE incr ( e_id NUMBER, amt NUMBER) IS
                vsalary NUMBER;
        salary_missing  EXCEPTION;
BEGIN
        SELECT sal INTO vsalary FROM emp WHERE  empno = e_id;
        IF vsalary IS NULL THEN
RAISE salary_missing;
        ELSE
                UPDATE emp SET sal = sal + amt WHERE empno = e_id;
        END IF;
EXCEPTION
                WHEN salary_missing THEN
                        dbms_output.put_line( e_id || 'has salary as NULL');
                WHEN NO_DATA_FOUND THEN
                        dbms_output.put_line( e_id|| 'no such NUMBER');
END incr;
/

SQL>SELECT * FROM emp WHERE empno=7902;
SQL>EXECUTE incr(7902,3000);
SQL> SELECT * FROM emp WHERE empno=7876;
SQL> EXECUTE incr(7876,1100);
SQL> SELECT * FROM emp WHERE empno=7876;
```

```
SQL> ed
Wrote file afiedt.buf

  1 CREATE or REPLACE PROCEDURE incr ( e_id NUMBER, amt NUMBER) IS
  2  vsalary NUMBER;
  3  salary_missing EXCEPTION;
  4  BEGIN
  5   SELECT sal INTO vsalary FROM emp WHERE empno = e_id;
  6   IF vsalary IS NULL THEN
  7  RAISE salary_missing;
  8   ELSE
  9   UPDATE emp SET sal = sal + amt WHERE empno = e_id;
 10   END IF;
 11  EXCEPTION
 12   WHEN salary_missing THEN
 13   dbms_output.put_line( e_id || 'has salary as NULL');
 14   WHEN NO_DATA_FOUND THEN
 15   dbms_output.put_line( e_id|| 'no such NUMBER');
```

```
 16* END incr;
SQL> /

Procedure created.
```

SQL>SELECT * FROM emp WHERE empno=7902;

```
SQL> select * from emp where empno=7902;

    EMPNO ENAME              JOB                  MGR HIREDATE
--------- -------------------- ------------------ --------- ---------
      SAL       COMM     DEPTNO
---------- ---------- ----------
     7902 ford             analyst              7566 04-DEC-81
     3000         20
```

SQL>EXECUTE incr(7902,3000);

```
SQL> execute incr(7902,3000);

PL/SQL procedure successfully completed.
```

SQL> SELECT * FROM emp WHERE empno=7876;

```
SQL> select * from emp where empno=7876;

    EMPNO ENAME              JOB                  MGR HIREDATE
--------- -------------------- ------------------ --------- ---------
      SAL       COMM     DEPTNO
---------- ---------- ----------
     7876 adams            clerk                7788 12-JAN-83
     1100         20
```

SQL> EXECUTE incr(7876,1100);

```
SQL> execute incr(7876,1100);

PL/SQL procedure successfully completed.
```

SQL> SELECT * FROM emp WHERE empno=7876;

SQL> select *from emp where empno=7876;

```
    EMPNO ENAME           JOB                 MGR HIREDATE
--------- ------------------ ------------------ --------- ---------
      SAL       COMM     DEPTNO
--------- ---------- ----------
     7876 adams             clerk               7788 12-JAN-83
     2200         20
```

FUNCTION

9B. CREATE A FUNCTION TO CALCULATE THE NET SALARY BASED ON VARIOUS CONDITIONS. THE EMPLOYEE NUMBER IS PARAMETER. THE FUNCTION RETURNS THE VALUE OF THE INCREMENT.

```
CREATE OR REPLACE FUNCTION review ( empid NUMBER)
        RETURN NUMBER IS
                incr            emp.sal%TYPE;
                net             emp.sal%TYPE;
                vempno          emp.empno%TYPE;
                vsal            emp.sal%TYPE;
                vcomm           emp.comm%TYPE;
BEGIN
        SELECT empno, sal, NVL(comm, 0) INTO vempno, vsal, vcomm FROM emp
                WHERE empno = empid;

        net := vsal + vcomm;
        IF vsal <= 3000 THEN
                incr := 0.20 * net;
         ELSEIF vsal > 3000 AND vsal <=6000 THEN
                incr :=0.30 * net;
         ELSE
                incr :=0.40 * net;
         END IF;
         RETURN (incr);
 END;
```

SQL> ed
Wrote file afiedt.buf

```
  1  CREATE OR REPLACE FUNCTION review ( empid NUMBER)
```

```
  2  RETURN NUMBER IS
  3   incr emp.sal%TYPE;
  4   net emp.sal%TYPE;
  5   vempno emp.empno%TYPE;
  6   vsal emp.sal%TYPE;
  7   vcomm emp.comm%TYPE;
  8  BEGIN
  9  SELECT empno, sal, NVL(comm, 0) INTO vempno, vsal, vcomm FROM
emp
 10  WHERE empno = empid;
 11  net := vsal + vcomm;
 12  IF vsal <= 3000 THEN
 13  incr := 0.20 * net;
 14  ELSIF vsal > 3000 AND vsal <=6000 THEN
 15   incr :=0.30 * net;
 16  ELSE
 17   incr :=0.40 * net;
 18  END IF;
 19  RETURN (incr);
 20* END;
SQL> /

Function created.
```

PL/SQL BLOCK CALLING REVIEW FUNCTION:

PL/SQL BLOCK CALLING REVIEW FUNCTION:

```
DECLARE
        incr_sal NUMBER(7,2);
BEGIN
        incr_sal := review(7698);
        dbms_output.put_line (incr_sal);

END;
/.
```

```
SQL> ed
Wrote file afiedt.buf

  1   DECLARE
  2   incr_sal NUMBER(7,2);
  3  BEGIN
  4   incr_sal := review(7698);
```

```
  5   dbms_output.put_line (incr_sal);
  6* END;
SQL> /
576

PL/SQL procedure successfully completed.
```

10A. CREATE A PACKAGE WITH FUNCTION TO INCREMENT THE SALARY AND RAISE ERROR MESSAGE IF ANY.

```
CREATE OR REPLACE PACKAGE emplpack AS

        PROCEDURE empproc ( empcode IN emp.empno%TYPE;);

        FUNCTION    increment ( e_id NUMBER, amt NUMBER) RETURN NUMBER;
END emplpack;

CREATE OR REPLACE PACKAGE BODY emplpack AS
        PROCEDURE          empproc ( empcode IN emp.empno%TYPE) IS
                           tempname emp.ename%TYPE;

        tesal emp.sal%TYPE;
        BEGIN
                SELECT ename, sal INTO tempname, tesal FROM emp WHERE empno
                        =empcode;

                dbms_output.put_line ( empcode||' '||tempname||' '||tesal);

                EXCEPTION
                        WHEN NO_DATA_FOUND THEN
                                dbms_out.put_line(empcode|| ' NOT FOUND');
        END empproc;
```

```
    FUNCTION increment (e_id NUMBER,     amt NUMBER) RETURN
                NUMBER IS vsal emp.sal%TYPE;

        sal_miss        EXCEPTION;
        temp            NUMBER;
    BEGIN
        SELECT sal INTO vsal FROM emp WHERE empno =e_id;

        IF vsal IS NULL THEN RAISE sal_miss;
        ELSE
            temp := vsal+amt;
            UPDATE emp SET sal = sal + temp WHERE empno = e_id;
            RETURN (temp);
        END IF;
        EXCEPTION
            WHEN sal_miss THEN
                dbms_output.put_line ( e_id || ' has salary as NULL');
            WHEN NO_DATA_FOUND THEN
                dbms_output.put_line( e_id || ' no such NUMBER');
    END increment;
END emplpack;
```

```
SQL> ed
Wrote file afiedt.buf

  1  CREATE OR REPLACE PACKAGE BODY emplpack AS
  2      PROCEDURE empproc ( empcode IN emp.empno%TYPE) IS
  3              tempname emp.ename%TYPE;
  4      tesal emp.sal%TYPE;
  5      BEGIN
  6              SELECT ename, sal INTO tempname, tesal FROM emp WHERE
empno=empcode;
  7      dbms_output.put_line (empcode||''||tempname||''||tesal);
  8      EXCEPTION
  9          WHEN NO_DATA_FOUND THEN
 10                  dbms_output.put_line(empcode||'NOT FOUND');
 11       END empproc;
 12       FUNCTION increment (e_id NUMBER, amt NUMBER) RETURN
 13            NUMBER IS vsal emp.sal%TYPE;
 14          sal_miss EXCEPTION;
 15          temp NUMBER;
 16       BEGIN
 17          SELECT sal INTO vsal FROM emp WHERE empno =e_id;
 18          IF vsal IS NULL THEN RAISE sal_miss;
 19          ELSE
 20              temp := vsal+amt;
```

```
 21              UPDATE emp SET sal = sal + temp WHERE empno = e_id;
 22              RETURN (temp);
 23          END IF;
 24          EXCEPTION
 25              WHEN sal_miss THEN
 26                  dbms_output.put_line ( e_id || ' has salary as NULL');
 27              WHEN NO_DATA_FOUND THEN
 28                  dbms_output.put_line( e_id || ' no such NUMBER');
 29      END increment;
 30* END emplpack;
SQL> /
```

Package body created.

PL/SQL block with package:

```
DECLARE
        incr_sal        NUMBER(7,2);
BEGIN
        emplpack.empproc(7777);
        emplpack.empproc(7698);
        incr_sal := emplpack.increment(7698,1000);
        emplpack.empproc(7698);
        dbms_output.put_line ('incremented sal is'||incr_sal);
END;
/
```

SQL> ed
Wrote file afiedt.buf

```
  1  DECLARE
  2   incr_sal NUMBER(7,2);
  3  BEGIN
  4   emplpack.empproc(7777);
  5   emplpack.empproc(7698);
  6   incr_sal := emplpack.increment(7698,1000);
  7   emplpack.empproc(7698);
  8   dbms_output.put_line ('incremented sal is'||incr_sal);
  9* END;
SQL> /
```

```
7777NOT FOUND
7698blake2850
7698blake6700
incremented sal is3850

PL/SQL procedure successfully completed.
```

10B. WRITE A TRIGGER TOTAL SAL TO MAINTAIN A DERIVED COLUMN TOTSAL THAT STORES TOTAL SALARY OF ALL MEMBERS IN A DEPARTMENT.

Procedure to Execute Trigger:

1.Alter a table dept with an additional column totasal

SQL>alter table dept add totsal VARHCAR(20);

```
SQL> alter table dept add totsal varchar(20);

Table altered.
```

2.UPDATE dept by totsal

SQL>UPDATE dept totsal=(SELECT sum(sal) FROM emp WHERE emp.deptno=dept.deptno);

```
SQL> update dept set totsal=(select sum(sal) from emp where
emp.deptno=dept.deptno);

4 rows updated.
```

SQL>SELECT * FROM, dept;

```
SQL> select * from dept;

    DEPTNO DNAME          LOC           TOTSAL
---------- -------------- ------------- --------------------
        10 accounting     new york
        20 research       dallas
        30 sales          chicago       4100
        40 operations     boston
```

3.Execute the program
SQL>/ Trigger CREATEd.

```
CREATE OR REPLACE TRIGGER total_salary
        AFTER DELETE OR INSERT OR UPDATE OF deptno , sal ON emp
        FOR EACH ROW

        BEGIN
                IF DELETING OR (UPDATING AND :old.deptno = :new.deptno) THEN
                        UPDATE dept SET totsal = totsal - :old.sal WHERE
                                deptno=:old.deptno;
                END IF;

                IF INSERTING OR (UPDATING AND :old.deptno = :new.deptno)
                        THEN
                                UPDATE dept SET totsal = totsal + :new.sal WHERE
                                deptno = :new.deptno;
                END IF;

                IF UPDATING AND :old.deptno = :new.deptno AND :old.sal != :new.sal
                        THEN
                                UPDATE dept SET totsal = totsal - :old.sal + :new.sal
                                WHERE deptno = :new.deptno;
                END IF;

                IF (:new.sal <500 OR :new.sal > 50000) THEN
                        RAISE_APPLICATION_ERROR( -20230, ' SALARY OUT OF
                                RANGE');
                END IF;
END;
```

```
SQL> ed
Wrote file afiedt.buf

  1  CREATE OR REPLACE TRIGGER total_salary
  2  AFTER DELETE OR INSERT OR UPDATE OF deptno , sal ON emp
  3  FOR EACH ROW
  4  BEGIN
  5   IF DELETING OR (UPDATING AND :old.deptno = :new.deptno) THEN
  6   UPDATE dept SET totsal = totsal - :old.sal WHERE
  7  deptno=:old.deptno;
  8   END IF;
  9  IF INSERTING OR (UPDATING AND :old.deptno = :new.deptno)
 10  THEN
 11   UPDATE dept SET totsal = totsal + :new.sal WHERE
```

```
 12  deptno = :new.deptno;
 13   END IF;
 14  IF UPDATING AND :old.deptno = :new.deptno AND :old.sal != :new.sal
 15  THEN
 16   UPDATE dept SET totsal = totsal - :old.sal + :new.sal
 17   WHERE deptno = :new.deptno;
 18  END IF;
 19  IF (:new.sal <500 OR :new.sal > 50000) THEN
 20  RAISE_APPLICATION_ERROR( -20230, ' SALARY OUT OF RANGE');
 21   END IF;
 22* END;
SQL> /

Trigger created.
```

SQL> DELETE FROM emp WHERE empno=7900;

```
SQL> delete from emp where empno=7900;

1 row deleted.
```

SQL> SELECT * FROM dept;

```
SQL> select * from dept;

    DEPTNO DNAME          LOC                TOTSAL
---------- -------------- ------------------ --------------------
        10 accounting     new york
        20 research       dallas
        30 sales          chicago            4100
        40 operations     boston
```

SQL> INSERT INTO emp VALUES(8000,'Karthik','dbaprgm',7782,'12-aug-08',800,500,30);

```
SQL> insert into emp values(8000,'Karthik','dbaprgm',7782,'12-aug-08',800,500,30);

1 row created.
```

SQL> SELECT * FROM dept;

```
SQL> select * from dept;

    DEPTNO DNAME              LOC                TOTSAL
---------- ------------------ ------------------ ------------------
        10 accounting         new york
        20 research           dallas
        30 sales              chicago            4900
        40 operations         boston
```

SQL> INSERT INTO emp VALUES(8000,'Karthik','dbaprgm',7782,'12-aug 08',8000,500,30);
INSERT INTO emp VALUES(8000,'Karthik','dbaprgm',7782,'12-aug-08',8000,500,30)
*****

```
SQL> insert into emp values(8000,'Karthik','dbaprgm',7782,'12-aug-08',8000,500,30);

1 row created.
```

SQL> UPDATE emp SET sal=800 WHERE empno=7876;

```
SQL> update emp set sal=800 where empno=7876;

1 row updated.
```

SQL> SELECT * FROM dept;

```
SQL> select * from dept;

    DEPTNO DNAME              LOC                TOTSAL
---------- ------------------ ------------------ ------------------
        10 accounting         new york
        20 research           dallas
        30 sales              chicago            12900
        40 operations         boston
```