PRACTICAL – 09

**NAME : Aryan Kashikar  
Roll n0: D2 32**

Practical 9: Procedure and functions

AIM: To implement stored procedures and function using PL/SQL on the database table for the underlying database.

Use table EMP and DEPT (scott)

1. Write procedure to accept data (for dept table) from user, store it into local

variables and insert data into table dept.

CREATE OR REPLACE PROCEDURE insert\_dept\_data(

    dept\_id\_in IN NUMBER,

    dept\_name\_in IN VARCHAR2,

    dept\_location\_in IN VARCHAR2

) AS

BEGIN

    INSERT INTO DEPT(DEPTNO, DNAME, LOC)

    VALUES(dept\_id\_in, dept\_name\_in, dept\_location\_in);

    COMMIT;

    DBMS\_OUTPUT.PUT\_LINE('Data inserted successfully.');

EXCEPTION

    WHEN OTHERS THEN

        DBMS\_OUTPUT.PUT\_LINE('Error: ' || SQLERRM);

        ROLLBACK;

END;

2. Use the EMP/DEPT Tables and create a stored procedure get\_salary with input parameter

e\_id, and output parameters employee name, department name and salary of the employee

with that id. Then write an anonymous PL/SQL program to execute this procedure with e\_id

= 2 and print out the output parameters employee name, department name and salary;

CREATE OR REPLACE PROCEDURE get\_salary(

    e\_id IN EMP.EMPNO%TYPE,

    emp\_name OUT EMP.ENAME%TYPE,

    dept\_name OUT DEPT.DNAME%TYPE,

    salary OUT EMP.SAL%TYPE

) AS

BEGIN

    SELECT ENAME, DNAME, SAL INTO emp\_name, dept\_name, salary

    FROM EMP

    JOIN DEPT ON EMP.DEPTNO = DEPT.DEPTNO

    WHERE EMP.EMPNO = e\_id;

EXCEPTION

    WHEN NO\_DATA\_FOUND THEN

        DBMS\_OUTPUT.PUT\_LINE('Employee with ID ' || e\_id || ' not found.');

    WHEN OTHERS THEN

        DBMS\_OUTPUT.PUT\_LINE('Error: ' || SQLERRM);

END;

3. Write a procedure to update salary in emp table for given empno.

(Hint pass take empno as input from user and pass it to procedure , procedure will perform

update

PROCEDURE sal\_raise (emp\_id IN NUMBER, sal\_incr IN NUMBER))

CREATE OR REPLACE PROCEDURE sal\_raise(

    emp\_id IN NUMBER,

    sal\_incr IN NUMBER

) AS

BEGIN

    UPDATE EMP

    SET SAL = SAL + sal\_incr

    WHERE EMPNO = emp\_id;

    COMMIT;

    DBMS\_OUTPUT.PUT\_LINE('Salary updated successfully.');

EXCEPTION

    WHEN NO\_DATA\_FOUND THEN

        DBMS\_OUTPUT.PUT\_LINE('Employee with ID ' || emp\_id || ' not found.');

    WHEN OTHERS THEN

        DBMS\_OUTPUT.PUT\_LINE('Error: ' || SQLERRM);

        ROLLBACK;

END;

4. The procedure deletes the employee with an employee number that corresponds to the

argument EMP\_ID. If no employee is found, an exception is raised.

(Hint use IF SQL%NOTFOUND THEN

raise\_application\_error(-20011, 'Invalid Employee’)

CREATE OR REPLACE PROCEDURE delete\_employee(

    emp\_id IN NUMBER

) AS

BEGIN

    DELETE FROM EMP

    WHERE EMPNO = emp\_id;

    IF SQL%NOTFOUND THEN

        raise\_application\_error(-20011, 'Invalid Employee');

    END IF;

    COMMIT;

    DBMS\_OUTPUT.PUT\_LINE('Employee deleted successfully.');

EXCEPTION

    WHEN OTHERS THEN

        DBMS\_OUTPUT.PUT\_LINE('Error: ' || SQLERRM);

        ROLLBACK;

END;

5. Write procedure to calculate rent using data of table vehiclerent .

[ refer to PDF shared in classroom]

CREATE TABLE VEHICLERENT ( VEHICLE\_ID NUMBER(8,0) PRIMARY KEY,

MANUFACTURER VARCHAR2(75) NOT NULL, MODELNAME VARCHAR2(20),

TYPE VARCHAR2(10), RENT\_DATE DATE, RETURN\_DATE DATE );

INSERT INTO VEHICLERENT VALUES(1, 'Honda', 'City', 'Car',TO\_DATE('2018-02-

20','YYYY-MM-DD'), TO\_DATE('2018-02-23','YYYY-MM-DD'));

INSERT INTO VEHICLERENT VALUES(2, 'Hyundai', 'i20', 'Car',TO\_DATE('2018-03-

10','YYYY-MM-DD'), TO\_DATE('2018-03-13','YYYY-MM-DD'));

CREATE PROCEDURE calculateRent(vehicleId IN NUMBER, rentAmount OUT

NUMBER)

CREATE OR REPLACE PROCEDURE calculateRent(

    vehicleId IN NUMBER,

    rentAmount OUT NUMBER

) AS

BEGIN

    SELECT RETURN\_DATE - RENT\_DATE INTO rentAmount

    FROM VEHICLERENT

    WHERE VEHICLE\_ID = vehicleId;

EXCEPTION

    WHEN NO\_DATA\_FOUND THEN

        DBMS\_OUTPUT.PUT\_LINE('Vehicle with ID ' || vehicleId || ' not found.');

    WHEN OTHERS THEN

        DBMS\_OUTPUT.PUT\_LINE('Error: ' || SQLERRM);

END;

6. Write a function to compute factorial of number. Function will take number as INPUT and

return factorial of number

CREATE OR REPLACE FUNCTION factorial(num IN NUMBER) RETURN NUMBER IS

    result NUMBER := 1;

BEGIN

    IF num = 0 THEN

        RETURN 1;

    ELSE

        FOR i IN 1..num LOOP

            result := result \* i;

        END LOOP;

        RETURN result;

    END IF;

END;

7. Write function to calculate average salary of deptno passed to function as In parameter.

(Hint CREATE FUNCTION average\_salary (e\_deptno IN INTEGER))

CREATE OR REPLACE FUNCTION average\_salary(e\_deptno IN INTEGER) RETURN NUMBER IS

    avg\_sal NUMBER;

BEGIN

    SELECT AVG(SAL) INTO avg\_sal

    FROM EMP

    WHERE DEPTNO = e\_deptno;

    RETURN avg\_sal;

END;

8. Define and Invoke a simple PL/SQL function which will compute and return the maximum of

three values.

CREATE OR REPLACE FUNCTION max\_of\_three(

    num1 IN NUMBER,

    num2 IN NUMBER,

    num3 IN NUMBER

) RETURN NUMBER IS

    max\_val NUMBER;

BEGIN

    max\_val := GREATEST(num1, num2, num3);

    RETURN max\_val;

END;