**\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*PLSQL\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\***

**LOOP\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\***

SQL> DECLARE

BEGIN

FOR I IN 1..5

LOOP

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

END LOOP;

END;

**PLSQL Lab -1.1.1\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\***

Identify the problems(if any) in the below declarations:

DECLARE

V\_Sample1 NUMBER(2);

V\_Sample2 CONSTANT NUMBER(2) ;

V\_Sample3 NUMBER(2) NOT NULL ;

V\_Sample4 NUMBER(2) := 50;

V\_Sample5 NUMBER(2) DEFAULT 25;

**Ans:**

DECLARE

v\_sample NUMBER(2);

v\_sample2 CONSTANT Number(2):=90;

v\_sample3 NUMBER(2) NOT NULL :=80;

v\_sample4 NUMBER(2) :=50;

v\_sample NUMBER(2) DEFAULT 25;

BEGIN

DBMS\_OUTPUT.PUT\_LINE('Hello');

END;

**PLSQL-Lab 1.1.2\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\***

The following PL/SQL block is incomplete.

Modify the block to achieve requirements as stated in the comments in the block.

DECLARE --outer block

var\_num1 NUMBER := 5;

BEGIN

DECLARE --inner block

var\_num1 NUMBER := 10;

BEGIN

DBMS\_OUTPUT.PUT\_LINE('Value for var\_num1:' ||var\_num1);

--Can outer block variable (var\_num1) be printed here.If Yes,Print the same.

END;

--Can inner block variable(var\_num1) be printed here.If Yes,Print the same.

END;

DECLARE

var\_num1 NUMBER:=5;

BEGIN

DECLARE

var\_num2 NUMBER:=10;

BEGIN

DBMS\_OUTPUT.PUT\_LINE('Value For var\_num1:'||var\_num1);

END;

/\* DBMS\_OUTPUT.PUT\_LINE('Value for var\_num2:'||var\_num2); not accessible \*/

END;

**PLSQL-Lab 1.1.3\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\***

Write a PL/SQL block to retrieve all staff (code, name, salary) under specific

department number and display the result. (Note: The Department\_Code will be accepted

from user. Cursor to be used.**)**

**Ans:-**

DECLARE

v\_dept\_code staff\_masters.dept\_code%TYPE;

v\_staff\_detail staff\_masters%ROWTYPE;

TYPE r\_staff\_rec IS RECORD(

v\_sf\_code staff\_masters.staff\_code%TYPE,

v\_sf\_nm staff\_masters.staff\_name%TYPE,

v\_sf\_sal staff\_masters.staff\_sal%TYPE );

v\_staff\_rec r\_staff\_rec;

CURSOR c\_staff\_details IS SELECT staff\_code,staff\_name,staff\_sal

FROM staff\_masters WHERE dept\_code=&v\_dept\_code;

BEGIN

IF NOT c\_staff\_details%ISOPEN THEN

OPEN c\_staff\_details;

END IF;

LOOP

FETCH c\_staff\_details INTO v\_staff\_rec;

DBMS\_OUTPUT.PUT\_LINE(v\_staff\_rec.v\_sf\_code||' '||v\_staff\_rec.v\_sf\_nm||' '||v\_staff\_rec.v\_sf\_sal);

EXIT WHEN c\_staff\_details%NOTFOUND;

END LOOP;

CLOSE c\_staff\_details;

END;

SQL> /

Enter value for v\_dept\_code: 20

old 11: FROM staff\_masters WHERE dept\_code=&v\_dept\_code;

new 11: FROM staff\_masters WHERE dept\_code=20;

100004 Anil 20000

100007 Smith 68200

100009 Rahul 22000

100009 Rahul 22000

PL/SQL procedure successfully completed.

**PLSQL Lab 1.1.4\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\***

Write a PL/SQL block to increase the salary by 30 % or 5000 whichever minimum for

a given Department\_Code.

DECLARE

v\_dept\_code staff\_masters.dept\_code%TYPE :=10;

v\_staff\_sal staff\_masters.staff\_sal%TYPE;

CURSOR c\_staff\_sal IS

SELECT staff\_code,staff\_name,staff\_sal FROM staff\_masters

WHERE dept\_code=v\_dept\_code FOR

UPDATE OF staff\_sal NOWAIT;

BEGIN

--if(not c\_staff\_sal%ISOPEN)THEN

--OPEN c\_staff\_sal;

---END IF;

FOR v\_rec IN c\_staff\_sal

LOOP

IF((v\_rec.staff\_sal\*0.3)>5000) THEN

UPDATE staff\_masters SET staff\_sal=((v\_rec.staff\_sal)+5000)

WHERE CURRENT OF c\_staff\_sal;

ELSE

UPDATE staff\_masters SET staff\_sal=(v\_rec.staff\_sal+(v\_rec.staff\_sal\*0.3))

WHERE CURRENT OF c\_staff\_sal;

END IF;

END LOOP;

END;

**PLSQL Lab 1.1.5\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\***

Write a PL/SQL block to generate the following report for a given Department code

Student\_Code Sudent\_Name Subject1 Subject2 Subject3 Total Percentage

Grade

Note: Display suitable error massage if wrong department code has entered and if there

is no student in the given department.

For Grade:

Student should pass in each subject individually (pass marks 60).

Percent >= 80 then grade= A

Percent >= 70 and < 80 then grade= B

Percent >= 60 and < 70 then grade= C

Else D

Ans:-

DECLARE

v\_deptno student\_masters.dept\_code%TYPE;

v\_total number(3);

v\_percentage int;

v\_grade VARCHAR2(1);

CURSOR c\_details IS

SELECT s.student\_code,s1.student\_name,s.subject1,s.subject2,s.subject3

FROM student\_marks s,

student\_masters s1

WHERE s.student\_code=s1.student\_code

AND s1.dept\_code=&v\_deptno;

BEGIN

FOR stu IN c\_details

LOOP

v\_total:=stu.subject1+stu.subject2+stu.subject3;

v\_percentage:=(v\_total/300)\*100;

IF v\_percentage>=80 THEN

v\_grade:='A';

ELSIF v\_percentage>=70 AND v\_percentage<80 THEN

v\_grade:='B';

ELSIF v\_percentage>=60 AND v\_percentage<70 THEN

v\_grade:='C';

ELSE

v\_grade:='D';

END IF;

DBMS\_OUTPUT.PUT\_LINE('STUDENT CODE: '||stu.student\_code||

' STUDENT NAME: '||stu.student\_name||

' SUBJECT1: '||stu.subject1||

' SUBJECT2: '||stu.subject2||

' SUBJECT3: '||stu.subject3||

' TOTAL: '||v\_total||

' PERCENTAGE:'||v\_percentage||

' GRADE: '||v\_grade);

END LOOP;

END;

**PLSQL Lab 1.1.5\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\***

Write a PL/SQL block to retrieve the details of the staff belonging to a particular

department. Department code should be passed as a parameter to the cursor.

Ans:

DECLARE

CURSOR c(v\_dept\_code staff\_masters.dept\_code%TYPE) IS

SELECT \* FROM staff\_masters

WHERE dept\_code=v\_dept\_code;

BEGIN

FOR staff IN c(&v\_dept\_code)

LOOP

DBMS\_OUTPUT.PUT\_LINE('staff code:'||staff.staff\_code||' '

||'staff name:'|| staff.staff\_name||' '

||'designation code:'||staff.design\_code||' '

||'departmentcode:'||staff.dept\_code||' '

||'staff salary: '||staff.staff\_sal||' '

||'hiredate: '||staff.hiredate||' '

||'address: '||staff.staff\_address);

END LOOP;

END;

**PLSQL-Lab 2.2.1\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\***

Modify the programs created in Lab2 to implement Exception Handling

DECLARE

v\_sal staff\_masters.staff\_sal%TYPE;

v\_bonus staff\_masters.staff\_sal%TYPE;

BEGIN

SELECT staff\_sal INTO v\_sal FROM staff\_masters WHERE mgr\_code=100006;

v\_bonus:=2\*v\_sal;

DBMS\_OUTPUT.PUT\_LINE('STAFF\_SALARY'||v\_sal);

DBMS\_OUTPUT.PUT\_LINE('Staff\_BONUS'||v\_BONUS);

EXCEPTION

WHEN NO\_DATA\_FOUND THEN

DBMS\_OUTPUT.PUT\_LINE('Given Code is not valid enter valid code');

WHEN TOO\_MANY\_ROWS THEN

DBMS\_OUTPUT.PUT\_LINE('MORE THAN ONE ROW');

WHEN VALUE\_ERROR THEN

DBMS\_OUTPUT.PUT\_LINE('truncation,arithmetic,conversion or constraints errors');

WHEN DUP\_VAL\_ON\_INDEX THEN

DBMS\_OUTPUT.PUT\_LINE('Unique constraints is violated');

END;

**Lab 2.2.2 \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\***

The following PL/SQL block attempts to calculate bonus of staff for a given

MGR\_CODE. Bonus is to be considered as twice of salary. Though Exception Handling

has been implemented but block is unable to handle the same.

Debug and verify the current behavior to trace the problem.

DECLARE

V\_BONUS V\_SAL%TYPE;

V\_SAL STAFF\_MASTER.STAFF\_SAL%TYPE;

BEGIN

SELECT STAFF\_SAL INTO V\_SAL

FROM STAFF\_MASTER

WHERE MGR\_CODE=100006;

V\_BONUS:=2\*V\_SAL;

DBMS\_OUTPUT.PUT\_LINE('STAFF SALARY IS ' || V\_SAL);

DBMS\_OUTPUT.PUT\_LINE('STAFF BONUS IS ' || V\_BONUS);

EXCEPTION

WHEN NO\_DATA\_FOUND THEN

DBMS\_OUTPUT.PUT\_LINE('GIVEN CODE IS NOT VALID.ENTER VALID CODE');

END;

**Ans:**

select \* from staff\_masters;

set serveroutput on;

DECLARE

v\_sal staff\_masters.staff\_sal%TYPE;

v\_bonus v\_sal%TYPE;

BEGIN

SELECT staff\_sal INTO v\_sal

FROM staff\_masters WHERE mgr\_code=&mgr\_code;

v\_bonus:=2\*v\_sal;

DBMS\_OUTPUT.PUT\_LINE('old salary : '||v\_sal);

DBMS\_OUTPUT.PUT\_LINE('new salary : '||v\_bonus);

EXCEPTION

WHEN too\_many\_rows THEN

DBMS\_OUTPUT.PUT\_LINE('there are too many rows with this mgr code');

END;

**Lab 2.2.3 \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\***

Rewrite the above block to achieve the requirement.

DECLARE

CURSOR emp\_cur IS SELECT \* FROM staff\_masters WHERE mgr\_code=&mgr\_code;

v\_bonus staff\_masters.staff\_sal%TYPE;

temp NUMBER;

invalid\_mgr EXCEPTION;

emp\_rec emp\_cur%ROWTYPE;

BEGIN

OPEN emp\_cur;

SELECT COUNT(\*) into temp FROM staff\_masters WHERE mgr\_code=&mgr\_code;

IF temp=0 THEN

RAISE invalid\_mgr;

ELSE

LOOP

FETCH emp\_cur INTO emp\_rec;

EXIT WHEN emp\_cur%NOTFOUND;

v\_bonus:=2\*emp\_rec.staff\_sal;

DBMS\_OUTPUT.PUT(emp\_rec.staff\_name||' ');

DBMS\_OUTPUT.PUT(emp\_rec.staff\_sal||' ');

DBMS\_OUTPUT.PUT\_LINE(v\_bonus);

END LOOP;

END IF;

CLOSE emp\_cur;

EXCEPTION

WHEN invalid\_mgr THEN

DBMS\_OUTPUT.PUT\_LINE('invalid mgr code');

END;

**Lab 2.2.4 \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\***

Predict the output of the following block ? What corrections would be needed to make it

more efficient?

BEGIN

DECLARE

fname emp.ename%TYPE;

BEGIN

SELECT ename INTO fname

FROM emp

WHERE 1=2;

DBMS\_OUTPUT.PUT\_LINE('This statement will print');

EXCEPTION

WHEN OTHERS THEN

DBMS\_OUTPUT.PUT\_LINE('Some inner block error');

END;

EXCEPTION

WHEN NO\_DATA\_FOUND THEN

DBMS\_OUTPUT.PUT\_LINE('No data found in fname');

WHEN OTHERS THEN

DBMS\_OUTPUT.PUT\_LINE('Some outer block error');

END;

**ANS:-**

BEGIN

DECLARE

fname emp.ename%TYPE;

BEGIN

SELECT ename into fname FROM emp WHERE 1=2;

DBMS\_OUTPUT.PUT\_LINE('line A');

EXCEPTION WHEN others THEN

DBMS\_OUTPUT.PUT\_LINE('Inner Begin error!!!!!');

END;

EXCEPTION WHEN no\_data\_found THEN

DBMS\_OUTPUT.PUT\_LINE('no data found!!!!!');

WHEN others THEN

DBMS\_OUTPUT.PUT\_LINE('other error!!!!!');

END;

\*\*\*\*\*\*\*\*OUTPUT\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

Inner Begin error!!!!!   
PL/SQL procedure successfully completed.

**Lab 2.2.5\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\***

Debug the above block to trace the flow of control.

Additionally one can make appropriate changes in Select statement defined in the

block to check the flow.

**Ans:-**

BEGIN

DECLARE

fname emp.ename%TYPE;

BEGIN

SELECT ename into fname FROM emp WHERE empno=&empno;

DBMS\_OUTPUT.PUT\_LINE('line A');

EXCEPTION WHEN others THEN

DBMS\_OUTPUT.PUT\_LINE('error!!!!!');

END;

EXCEPTION

WHEN no\_data\_found THEN

DBMS\_OUTPUT.PUT\_LINE('no data found!!!!!');

WHEN others THEN

DBMS\_OUTPUT.PUT\_LINE('other error!!!!!');

END;

Output………………………..

old 5: SELECT ename into fname FROM emp WHERE empno=&empno;   
new 5: SELECT ename into fname FROM emp WHERE empno=7839;   
line A   
PL/SQL procedure successfully completed.

**PLSQL Lab 2.2.6\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\***

Write a PL/SQL program to check for the commission for an employee no 7369. If no

commission exists, then display the error message. Use Exceptions.

**PLSQL Lab 2.2.7\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\***

Write a PL/SQL block to drop any user defined table.

**PLSQL Lab 3.1\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\***

DECLARE

v\_staff\_code staff\_masters.staff\_code%TYPE;

v\_staff\_nm staff\_masters.staff\_name%TYPE;

v\_staff\_sal staff\_masters.staff\_sal%TYPE;

v\_dept\_code staff\_masters.dept\_code%TYPE;

CURSOR c\_staff\_details IS

SELECT staff\_code,staff\_name,staff\_sal

FROM staff\_masters

WHERE dept\_code=&v\_dept\_code;

no\_row\_exe EXCEPTION;

BEGIN

IF NOT c\_staff\_details%ISOPEN THEN

OPEN c\_staff\_details;

END IF;

FETCH c\_staff\_details INTO v\_staff\_code,v\_staff\_nm,v\_staff\_sal ;

IF (c\_staff\_details%NOTFOUND) THEN

RAISE no\_row\_exe;

END IF;

LOOP

DBMS\_OUTPUT.PUT\_LINE(

' CODE: ' ||v\_staff\_code||

' Name:' ||v\_staff\_nm||

' Salary:' ||v\_staff\_sal);

FETCH c\_staff\_details INTO v\_staff\_code,v\_staff\_nm,v\_staff\_sal;

EXIT WHEN c\_staff\_details%NOTFOUND;

END LOOP;

EXCEPTION

WHEN no\_row\_exe THEN

DBMS\_OUTPUT.PUT\_LINE(v\_dept\_code ||'dept code does not exist or is having no rows');

END;

result--

SQL> /

Enter value for v\_dept\_code: 20

old 9: WHERE dept\_code=&v\_dept\_code;

new 9: WHERE dept\_code=20;

CODE: 100004 Name:Anil Salary:40000

CODE: 100007 Name:Smith Salary:62000

CODE: 100009 Name:Rahul Salary:22000

PL/SQL procedure successfully completed.

/

**PLSQL Lab 4.9\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\***

4.9. Write a package which will contain a procedure and a function.

Function: This function will return years of experience for a staff. This function will take

the hiredate of the staff as an input parameter. The output will be rounded to the nearest

year (1.4 year will be considered as 1 year and 1.5 year will be considered as 2 year).

Procedure: Capture the value returned by the above function to calculate the additional

allowance for the staff based on the experience.

Additional Allowance = Year of experience x 3000

Calculate the additional allowance and store Staff\_Code, Date of Joining, and Experience

in years and additional allowance in Staff\_Allowance table.

Ans:-

DROP TABLE staff\_allow;

CREATE TABLE STAFF\_allow(staff\_code NUMBER(30),hiredate DATE,

exp NUMBER(10),staff\_all NUMBER(30,3));

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

show errors;

CREATE OR REPLACE PACKAGE pkg1 AS

v\_temp VARCHAR2(10);

FUNCTION calcStaffExp(vi\_hiredate IN VARCHAR2) RETURN NUMBER;

PROCEDURE calcIncetive(i\_sc IN staff\_masters.staff\_code%TYPE);

END pkg1;

set serveroutput on;

show errors;

CREATE OR REPLACE PACKAGE BODY pkg1 AS

FUNCTION calcStaffExp(vi\_hiredate IN VARCHAR2) RETURN NUMBER AS

v\_hiredate staff\_masters.hiredate%TYPE;

v\_exp NUMBER;

BEGIN

v\_hiredate:=TO\_DATE(vi\_hiredate );

SELECT ROUND (MONTHS\_BETWEEN(SYSDATE,v\_hiredate)/12) INTO v\_exp FROM DUAL;

return v\_exp;

EXCEPTION

WHEN others THEN

DBMS\_OUTPUT.PUT\_LINE(' Some Exception');

END calcStaffExp;

-- Procedure Defination...............

PROCEDURE calcIncetive(i\_sc IN staff\_masters.staff\_code%TYPE)

AS

vp\_exp NUMBER;

v\_allowance NUMBER(30);

v\_hd staff\_masters.hiredate%TYPE;

v\_sc staff\_masters.staff\_code%TYPE;

BEGIN

SELECT hiredate INTO v\_hd FROM staff\_masters

WHERE staff\_code=i\_sc;

vp\_exp:=calcStaffExp(TO\_CHAR(v\_hd));

DBMS\_OUTPUT.PUT\_LINE(' EXP is '||vp\_exp);

v\_allowance:=vp\_exp\*3000;

DBMS\_OUTPUT.PUT\_LINE(' Allowlance is '|| v\_allowance);

INSERT INTO STAFF\_allow VALUES( v\_sc, v\_hd ,vp\_exp,v\_allowance);

commit;

END calcIncetive;

END pkg1;

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

set serveroutput on;

exec pkg1.calcIncetive(100002);

**Model Paper SQL\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\***

CREATE TABLE customer(customer\_code NUMBER PRIMARY KEY, title CHAR(3),

name VARCHAR2(20),surname VARCHAR2(20), phone\_number NUMBER(10), email VARCHAR2(30));

INSERT INTO customer VALUES(102,'Mr','Aakash','S',9866788558,'akash@yahoo.com');

INSERT INTO customer VALUES(101,'Mr','Preetham','K',9888788888,'preetham@gmail.com');

INSERT INTO customer VALUES(103,'Ms','Reena','M',9743788558,'reena@yahoo.com');

INSERT INTO customer VALUES(104,'Mr','Kailash',NULL,9437988558,'kailash@gmail.com');

1. Write a query which will find the number of customers who had entered their surname. [Marks 5]

2. INSERT 3 records into the feedback table. Use sequence to generate the feedback\_code [Marks 5]

3. Write a query which will display the name of the customer and the feedback given by the customers [Marks 5]

4. Write a procedure which will accept the customer code and return the phone number and email address of the customer. [Marks 15]

Note: Take care of necessary validation and exception handling

CREATE OR REPLACE PROCEDURE getCustDetails(i\_cust\_id NUMBER,

o\_phoneno OUT NUMBER, o\_emailadd OUT VARCHAR2) IS

e\_negative\_number EXCEPTION;

BEGIN

IF(i\_cust\_id<0) THEN

RAISE e\_negative\_number;

END IF;

SELECT email,phone\_number INTO

o\_emailadd,o\_phoneno FROM customer

WHERE customer\_code =i\_cust\_id;

EXCEPTION

WHEN NO\_DATA\_FOUND THEN

RAISE\_APPLICATION\_ERROR(-20045,'No Customer Id');

WHEN e\_negative\_number THEN

RAISE\_APPLICATION\_ERROR(-20046,'Negatibve number');

END;

DECLARE

v\_e VARCHAR2(100);

v\_p NUMBER;

BEGIN

getCustDetails(102,v\_p,v\_e);

DBMS\_OUTPUT.PUT\_LINE( v\_e || v\_p);

END;

/

variable email varchar2(20);

variable pho NUMBER;

print email ;

..............................................

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*SQL-99-JOIN On 3 Tables\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

SELECT \* from T1 Join T2 On C1 JOIN T3 On C2;