Practical 1

```
DROP TABLE emp;
DROP TABLE project;
*/
CREATE TABLE project (pno NUMBER(3)
PRIMARY KEY, pname VARCHAR2(20), thrs
NUMBER(4), super_no NUMBER(2));
CREATE TABLE emp (eno NUMBER(3)
PRIMARY KEY, ename VARCHAR2(25), hrs
NUMBER(2), pno NUMBER(2) REFERENCES
project(pno) , super_no NUMBER(2));

/* A. Insert Trigger */
CREATE OR REPLACE TRIGGER
emp_insert_trigger
AFTER INSERT ON emp
FOR EACH ROW
WHEN(NEW.pno IS NOT NULL)
BEGIN
UPDATE project SET thrs = thrs +
:NEW.hrs WHERE pno = :NEW.pno;
END;
/

/* B. Update Trigger */
CREATE OR REPLACE TRIGGER
emp_update_trigger
AFTER UPDATE ON emp
```

```
FOR EACH ROW
WHEN(NEW.pno IS NOT NULL)
BEGIN
UPDATE project SET thrs = thrs - :OLD.hrs
+ :NEW.hrs WHERE pno = :NEW.pno;
END;
/

/* C. Delete Trigger */
CREATE OR REPLACE TRIGGER
emp_delete_trigger
BEFORE DELETE ON emp
FOR EACH ROW
WHEN(OLD.pno IS NOT NULL)
BEGIN
UPDATE project SET thrs = thrs - :OLD.hrs
WHERE pno = :OLD.pno;
END;
/

/* Main */
INSERT INTO project VALUES(1, 'College
Website', 0, 1);
INSERT INTO project VALUES(2, 'Travels
Website', 0, 1);
INSERT INTO project VALUES(3,
'Tourism Website', 0, 1);
INSERT INTO emp VALUES(1, 'Drake', 5,
1, 1);
```

```
INSERT INTO emp VALUES(2, 'Josh' , 6,
1, 1);
INSERT INTO emp VALUES(3, 'John' , 5,
2, 1);
INSERT INTO emp VALUES(4, 'Arun' , 8,
3, 2);
SELECT * FROM project;
UPDATE emp SET hrs = 6 WHERE eno = 1;
SELECT * FROM project;
DELETE FROM emp WHERE eno = 3;
SELECT * FROM project;
```

Practical 2

```
DROP TABLE supplier;
DROP SEQUENCE practical2_sequence;
*/
CREATE TABLE supplier(supid NUMBER(5)
PRIMARY KEY, suppname VARCHAR2(15));
CREATE SEQUENCE practical2_sequence
START WITH 1 MAXVALUE 100;
BEGIN
 LOOP
 INSERT INTO supplier VALUES(practical2_sequence.NEXTVAL, 1);
 EXIT WHEN practical2_sequence.CURRVAL >= 100;
END LOOP;
END;
/
SELECT * FROM supplier;
```

Practical 3

```
DROP TABLE emp;
DROP SEQUENCE practical3_sequence;
*/
CREATE TABLE emp(eno NUMBER(5), ename VARCHAR2(15));
CREATE SEQUENCE practical3_sequence START WITH 10 CYCLE MINVALUE 10
MAXVALUE 50 INCREMENT BY 10 CACHE 2;
BEGIN
FOR i IN 1..10 LOOP
 INSERT INTO emp VALUES(practical3_sequence.NEXTVAL, 'Name');
END LOOP;
END;
/
SELECT * FROM emp;
```

Practical 4

```
DROP TABLE emp;
*/
CREATE TABLE emp (eno NUMBER(3) PRIMARY KEY, ename VARCHAR2(25),
salary NUMBER(6), join_date DATE);
INSERT INTO emp VALUES(1, 'Drake', 15000, TO_DATE('19990505','YYYYMMDD'));
INSERT INTO emp VALUES(2, 'Josh' , 15000,
TO_DATE('20001212','YYYYMMDD'));
INSERT INTO emp VALUES(3, 'John' , 20000,
TO_DATE('20020805','YYYYMMDD'));
INSERT INTO emp VALUES(4, 'Luke' , 10000,
TO_DATE('20020806','YYYYMMDD'));

DECLARE
CURSOR emp_cursor IS SELECT eno, salary, join_date FROM emp;
v_eno emp.eno%TYPE;
v_salary emp.salary%TYPE;
v_join_date
emp.join_date%TYPE;

BEGIN
OPEN emp_cursor;
LOOP
 FETCH emp_cursor INTO v_eno, v_salary, v_join_date;
 IF
emp_cursor%NOTFOUND THEN
 EXIT;
 ELSIF v_join_date = TO_DATE('20001212','YYYYMMDD')
THEN UPDATE
```

```
emp SET salary = v_salary*1.15 WHERE eno = v_eno;
 ELSE   UPDATE
emp SET salary = v_salary*1.05 WHERE eno= v_eno;
 END IF;
END LOOP;
CLOSE emp_cursor;
END;
/
SELECT * FROM emp;
```

Practical 5

```
DROP TABLE lecturer;
*/
SET SERVEROUTPUT ON
CREATE TABLE lecturer (lid NUMBER(4) PRIMARY KEY, lname VARCHAR2(14),
majorsubject VARCHAR2(10));
INSERT INTO lecturer VALUES(1, 'Rajesh', 'CS');
INSERT INTO lecturer VALUES(2, 'Jayesh', 'IT');
INSERT INTO lecturer VALUES(3, 'Suresh', 'CS');
INSERT INTO lecturer VALUES(4, 'Ramesh', 'BT');
/*INPUT CAN BE 1,2,3,4*/
DECLARE
v_lid lecturer.lid%TYPE := &v_lid;v_lname lecturer.lname%TYPE;v_majorsubject
lecturer.majorsubject%TYPE;
BEGIN
SELECT lname, majorsubject INTO v_lname, v_majorsubject FROM
lecturer WHERE lid = v_lid;
CASE v_majorsubject
 WHEN 'CS' THEN DBMS_OUTPUT.PUT_LINE(v_lname || ' : Computer Science');
 WHEN 'IT' THEN DBMS_OUTPUT.PUT_LINE(v_lname || ' : Information Technology');
 WHEN 'BT' THEN DBMS_OUTPUT.PUT_LINE(v_lname || ' : Biotechnology');
END CASE;
END;
/
```

Practical 6

```
DROP TABLE employee;
DROP TABLE department;
*/
SET SERVEROUTPUT ON
CREATE TABLE department(deptno NUMBER(4) PRIMARY KEY, dname VARCHAR2(40));
INSERT INTO department VALUES(1, 'Computer Science');
INSERT INTO department VALUES(2, 'Information Technology');
INSERT INTO department VALUES(3, 'Microbiology');
CREATE TABLE employee (empno NUMBER(4) PRIMARY KEY, ename VARCHAR2(10),
 salary NUMBER(10), deptnoNUMBER(4) REFERENCES department(deptno));
INSERT INTO employee VALUES(1, 'Drake', 15000, 1);
INSERT INTO employee VALUES(2, 'Josh', 15000, 1);
INSERT INTO employee VALUES(3, 'John', 20000, 2);
INSERT INTO employee VALUES(4, 'Luke' , 10000, 2);
INSERT INTO employee VALUES(5, 'Mike', 10000, 2);
INSERT INTO employee VALUES(6, 'Jazz', 16000, 3);
INSERT INTO employee VALUES(7, 'Salim', 20000, 1);
INSERT INTO employee VALUES(8, 'Kali' ,30000, 3);
INSERT INTO employee VALUES(9, 'Ramesh' , 11000, 1);
INSERT INTO employee VALUES(10, 'Suresh' , 21000, 1);
INSERT INTO employee VALUES(11, 'Kavitha' , 15000, 2);
INSERT INTO employee VALUES(12, 'Gayathri' , 15000, 3);
INSERT INTO employee VALUES(13, 'Kamal' , 37000, 2);
INSERT INTO employee VALUES(14, 'Nitish' , 89000, 1);
INSERT INTO employee VALUES(15, 'Hitesh' , 89000, 3);
INSERT INTO employee VALUES(16, 'Ajit' , 56000, 2);
```

```
/*A. Using While Loop*/
DECLARE
  CURSOR department_cursor IS
  SELECT deptno, dname FROM department;
  v_deptno   department.deptno%TYPE;
  v_dname   department.dname%TYPE;
  v_salary_sum NUMBER(10);
  v_employee_count NUMBER(3);
BEGIN
  OPEN department_cursor;
  FETCH department_cursor
 INTO
  v_deptno, v_dname;
WHILE
  department_cursor%FOUND LOOP
  SELECT
  COUNT(empno), SUM(salary)
INTO
  v_employee_count, v_salary_sum
FROM
  employee WHERE deptno = v_deptno;

DBMS_OUTPUT.PUT_LINE(v_deptno || ' ' || v_dname || ' Total Salary: ' ||
    v_salary_sum || ' Employee Count:' || v_employee_count);
 FETCH
department_cursor
INTO v_deptno, v_dname;
END LOOP;
```

```
CLOSE department_cursor;
END;
/

/*B. Using For Loop*/
DECLARE
CURSOR department_cursor IS
SELECT * FROM department;v_salary_sum NUMBER(10);
v_employee_count NUMBER(3);
v_department_row
department%ROWTYPE;
BEGIN
  FOR v_department_row IN
 department_cursor LOOP
SELECT
 COUNT(empno), SUM(salary) INTO
 v_employee_count, v_salary_sum FROM
employee WHERE deptno = v_department_row.deptno;

DBMS_OUTPUT.PUT_LINE(v_department_row.deptno || ' ' || v_department_row.dname ||
 ' Total Salary:' || v_salary_sum || ' Employee Count:' || v_employee_count);
END LOOP;
END;
/
```

PRATICAL 7

```
SET SERVEROUTPUT ON


DECLARE
  CURSOR employee_cursor IS SELECT * from employee;
  v_employee_row employee%ROWTYPE;
  v_dname department.dname%TYPE;


BEGIN
  FOR v_employee_row IN  employee_cursor LOOP
 IF
   v_employee_row.empno = 10 OR  v_employee_row.empno = 15 THEN
   GOTO skip;
 END IF;
 SELECT dname INTO
     v_dname FROM department WHERE deptno= v_employee_row.deptno;


DBMS_OUTPUT.PUT_LINE(v_employee_row.empno || ' ' || v_employee_row.ename || '' || v_dname);
 <<skip>>
 NULL;
END LOOP;
END;
/
```

Practical 8

```
SET SERVEROUTPUT ON

CREATE OR REPLACE PROCEDURE

increase_salary (p_empno employee.empno%TYPE) ISv_salary employee.salary%TYPE;

v_ename employee.ename%TYPE;

BEGIN

    SELECT ename, salary INTO v_ename, v_salary FROM employee WHERE empno = p_empno;

UPDATE employee SET salary = v_salary*1.15 WHERE empno = p_empno;

DBMS_OUTPUT.PUT_LINE('Incremented Salary for ' || v_ename || ' from ' || v_salary || ' to ' ||
v_salary * 1.15);

EXCEPTION

 WHEN

NO_DATA_FOUND THEN


DBMS_OUTPUT.PUT_LINE('Record Not Found');

END;

/


/* NOTE INPUT RANGES FROM 1-16*/

DECLARE

    v_empno employee.empno%TYPE := &v_empno;

BEGIN

increase_salary(v_empno);

END;

/


/* B. Select record of input employee id. */

CREATE OR REPLACE PROCEDURE

display_employee (p_empno employee.empno%TYPE)
```

```
IS
v_ename employee.ename%TYPE;
v_salary employee.salary%TYPE;
v_deptno employee.deptno%TYPE
;v_dname department.dname%TYPE;
BEGIN
   SELECT ename, salary,
 deptno INTO v_ename, v_salary, v_deptno FROM
   employee WHERE empno = p_empno;
SELECT dname INTO v_dname
FROM department WHERE deptno = v_deptno;
DBMS_OUTPUT.PUT_LINE(v_ename || ' ' || v_salary || ' ' || v_dname);
EXCEPTION
 WHEN NO_DATA_FOUND THEN
 DBMS_OUTPUT.PUT_LINE('Record Not
Found');
END;
/
/* NOTE INPUT RANGES FROM 1-16 */
DECLARE
v_empno employee.empno%TYPE := &v_empno;
BEGIN
display_employee(v_empno);
END;
/
```