Experiment - 1

Aim :- Write the queries for DDL and DML.

DDL (Data Definition Language) statements :

CREATE - to create objects in the database
ALTER - alters the structure of the database
DROP - delete objects from the database
TRUNCATE - remove all records from a table, including all spaces allocated for the records are removed.
COMMENT - add comments to the data dictionary.

DML (Data Manipulation Language) statements :

SELECT - retrieve data from the a database
INSERT - insert data into a table
UPDATE - updates existing data within a table
DELETE - delets all records from a table, the space for the records remain
CALL - call a PL/SQL or Java subprogram

DCL (Data Control Language)

GRANT - gives user's access privileges to database
REVOKE - withdraw access privileges given with the GRANT command.

Q1. Write SQL Query to create the schema for Employee Table

```
SQL > create table Employee (Fname varchar(20), Minit char, Lname varchar(30)
, Ssn int, Bdate date, Address varchar (50), Sex char, Salary int, Super_ssn
int, Dno int);
```

```
SQL > CREATE TABLE EMPLOYEE
(Fname VARCHAR(15) NOT NULL, Minit CHAR, Lname VARCHAR (15)
NOT NULL, Ssn CHAR (9) NOT NULL,
Bdate DATE, Address VARCHAR (30), Sex CHAR, Salary DECIMAL (10,2),
Super_ssn CHAR (9), Dno INT NOT NULL,
```

PRIMARY KEY (Ssn),
FOREIGN KEY (Super_ssn) REFERENCES EMPLOYEE (Ssn),
FOREIGN KEY (Dno) REFERENCES DEPARTMENT (Dnumber));

Q2. Write SQL Query to add the primary and foreign keys into the tables.

SQL> Alter table Employee add constraint fk FOREIGN KEY (Dno) REFERENCES DEPARTMENT (Dnumber)

Q3. Write SQL Query to delete primary key and foreign key.

Alter table employee drop constraint fk.

Q4. Write SQL Query to change data type of address to VARCHAR(60)

Alter table Employee modify Address varchar (60);

Q5. Write SQL Query to add new column named age in Employee table.

Alter table Employee add age int:

Q6. Write SQL Query to delete column named age in Employee table.

Alter table Employee drop column age:

Q.7. Write SQL Query to rename table Employee to Employee1.

Rename Employee to Employee1;

Q.8 Write SQL Query to insert values in Employee table:

Insert into employee values ('John', 'B', 'Smith', 123456789, '09-Jan-1965', '731 Fondren, Houston Tx', 'M', 30000, '', '');

Experiment - 2

Aim: - Write SQL queries using logical operations (=, <, >, etc).

The basic form of the SELECT statement, sometimes called a mapping or a select-from-where block, is formed of the three clauses SELECT, FROM, and WHERE and has the following form

SELECT < attribute list >
FROM < table list >
WHERE < condition >;

where

⇒ < attribute list > is a list of attribute names whose values are to be retrieved by the query.

⇒ < table list > is a list of the relation names required to process the query.

⇒ < condition > is a conditional (Boolean) expression that identifies the tuples to be retrieved by the query.

In SQL, the basic logical comparison operators for comparing attribute values with one another and with literal constants are =, <, <=, >, >=, and <>. These correspond to the relational algebra operators =, <, ≤, >, ≥ and ≠, respectively and to the C/C++ programming language operators =, <, <=, >, >= and !=

Q1. Write SQL Query to find the name of employee having salary greater than 20000.

SQL > Select fname from Employee where salary > 20000;

Q2. Write SQL Query to delete the record of employee whose salary is less than 10000.

SQL > delete from employee where salary < 10000;

Q3. Write SQL Query to increment the salary of employee by 10 percent.

SQL > update employee set salary = salary + salary * 0.1;

Q4. Write SQL Query to impose the constraint that the salary of each employee should be greater than 5000.

SQL > alter table employee add constraint ck1 check (salary > 5000);

Q5. Write SQL query to change the department no of John to 6.

SQL > update employee set dno = 6 where fname = 'John';

Q6. Write SQL query to delete the record of John.

SQL > delete from employee where fname = 'John';

Q.7. Write SQL query to display all the records of Employee table.

SQL > Select * from Employee;

Q10. Write SQL query to apply check constraint on Employee table.

SQL > Alter table Employee add constraint ck check (sex = 'M' or sex = 'F');

— T John

Experiment - 3

Aim :- Write SQL queries using SQL operators.

Like comparison can be used for string pattern matching. Partial strings are specified using two reserved characters: % replaces an arbitrary number of zero or more characters, and the underscore(_) replaces a single character. For example, consider the following query. Relational operators are {and, or} these are used selection of multiple condition.

Q1. Write SQL query to find out the name of employees living in Houston and having salary greater than 20000.

```
select fname from employee where address like '%Houston%'
and salary >20000;
```

Q2. Write SQL query to find out the name of female employees working in department 4 or 5.

```
select fname from employee where sex = 'f' and dno=4 or dno=5;
```

Q3. Write SQL query to decrease the salary of employees by 500 having salary less than 20000 and working in departments.

```
update employee set salary = salary +500 where salary <20000
and dno = 5;
```

Q4. Find SSN number of manager of research and Administration Department.

```
Select MGRSSn from Department where Dname = 'Research' or
Dname = 'Headquarters'; 'Administration';
```

Q5. Find the Joining date of manager of research and head-quarters department.

```
Select MGR_Start_Date from Department where Dname = 'Research'
or Dname = 'Headquarters';
```

Q6. Find the name of Employees Not working in department number 4.

```
Select Fname from Employee where dno <>4;
```

Experiment - 4

Aim :- Write SQL query using character, number, date and group functions.

Q1. Determine the maximum and minimum salaries. Rename the output as MAX_SAL and MIN_SAL respectively.
SQL > select min(salary) as Min_Sal, max(salary) as Max_Sal from employee;

Q2. Count the number of employees.
SQL > select count(ssn) from employees

Q3. Count the number of employees working in department number 5.
SQL > select count(*) from employee where dno = 5;

Q.4. Calculate the average salary of all employees.
SQL > select avg(salary) from employee;

Q5. Increase the birthdate of each employee by 4 months.
SQL > select ADD_MONTHS(bdate, 4) from employee;

Q6. Find the ascii value of the middle name of each employee.
SQL > select ASCII(Minit) from Employee;

Q7. Find the greatest value among 4, 5 and 17. Rename the output as G_NUM.
SQL > select GREATEST (4,5,17) "G_Num" from dual;

Q8. Retrieve the names of employees in upper case.
SQL > select Upper (fname) from Employee;

Q9. Retrieve the last four characters of the SSN of each employee.
SQL > select substr(ssn,6) from employee;

Q10. Retrieve the number of months between 02Feb1992 and 02Apr1992. Rename the output as Months.
SQL > select months_between ('02-feb-1992', '02-apr-1992') "Month from dual;

Q11. Count the number of employees having salary less than 40000.
SQL > select count(*) from employee where salary < 40000.

Experiment - 5

Aim :- Write SQL queries for relational algebra.

Q1. List all employee details along with the department name to which they belong.

SQL > select * from employee join department on dno = dnumber;

Q2. List the employee's ssn and name of only those employees who belong to the Administration department.

SQL> select ssn, fname, minit, lname from employee join department on dno = dnumber with dname = 'Administration';

Q3. List the full name, ssn along with the department name and number which they manage using left outer join operator.

SQL > select ssn, fname, minit, lname, dname, dnumber from employee left outer join department on ssn = mgr_ssn;

Q.4. List the full name, ssn of ~~department~~ employees who act both as a supervisor and supervise using right outer join.

SQL> select S.fname "Supervisor_Name" from employee E right outer join employee S on E.super_ssn = S.ssn;

Q.5. Find the birthdate and address of Ramesh K Narayan as well as Franklin T Wong.

SQL > select bdate, address from employee where fname = 'Franklin' ad and minit = 'T' and lname = 'Wong'
union
select bdate, address from employee where fname = 'Ramesh' and minit = 'K' and lname = 'Narayan';

Q6. List the department numbers where both James E Borg and Franklin T Wong are working as supervisors.

SQL > select E.dno from employee E, employee S where S.ssn = E.super_ssn and S.fname = 'James' and S.minit = 'E' and S.lname = 'Borg'
intersect
select E.dno from employee E, employee S where S.ssn = E.super_ssn and S.fname = 'Franklin' and S.minit = 'T' and S.minit = 'T' and S.lname = 'Wong'

Q7. Find the name of employees either working in research department or living in Houston, TX.

SQL > select fname, minit, lname from employee, department where dno = dnumber and dname = 'Research'
union
select fname, minit, lname from employee where address like '%Houston%';

Q.8 Find the salary and supervisor name of male and female employees having same supervisor and same salary.

SQL > select E.salary, E.super_ssn, S.fname from employee E, employee S where E.super_ssn = S.ssn and E.sex = 'F'
intersect
select E.salary, E.super_ssn, S.fname from employee E, employee S where E.super_ssn = S.ssn and E.sex = 'M';

Q9. Retrieve the name and number of department in which number of employees is less than 3 using inner join operator.

SQL > select dno, count(*), dname from employee join department on dno = dnumber group by dno, dname having count(*) >= 3;

Q10. Find the sum of salaries of all employees of the 'Research' department as well as the maximum salary, minimum salary and the average salary in this department.

SQL > select sum(salary) "SUM", min(salary) "MIN", max(salary) "MAX", avg(salary) "AVG" from employee, department where dno = dnumber and dname = 'Research';

Q11. For each employee, retrieve the employee's first and last name and the first and last name of his or her immediate supervisor, if any using left outer join.

SQL > select E.fname, E.lname, S.fname, S.lname from employee E left outer join employee S on E.super_ssn = S.ssn';

Q12. Count the number of distinct salary values in the database.

SQL > select count(DISTINCT salary) from employee;

Q.13 Find the name of manager of each department.

SQL > select fname, minit, lname, dname from employee, department where ssn = mgr_ssn;

Experiment-6

Aim:- Write SQL queries for extracting data from more than one table.

Q1. Write SQL query to find out the name of manager of each department.

SQL > select fname from employee, department where ssn=mgr_ssn;

Q2. Write SQL query to find out the name of female employees working in research department.

SQL > select fname from employee, department where sex = 'F' and Dnumber = dno;

Q.3. Write SQL query to find out the location of each department.

SQL > Select dlocation from department, dept_locations where department.dnumber = dept_locations.dnumber;

Q.4. Write SQL query to find the name of employees and their dependent name.

SQL > Select fname, Dependent_Name from Employee, Dependent where ssn = essn;

Q5. Write SQL query to find the name of employees working on project ProductX.

SQL > Select Fname from Employee, Works_on, Project where ssn = essn and pname = 'ProductX' and pno = Pnumber;

Q6. Write SQL query to find the name of employees working on project whose location is stafford.

SQL > Select Fname from Employee, Works_on, Project where ssn = essn and pname = 'ProductX' location = 'Stafford' and pno = Pnumber;

Q.7. Write SQL query to find the department name of John B Smith.

SQL > Select Dependent_name from Employee, Dependent where ssn = essn and fname = 'John' and Minit = 'B' = Lname = 'Smith'

Q8. Write SQL query to find the name of projects whose location or department in which John B Smith works in.

SQL > select fname from project, department, employee, dept_locations, works_on. where fname = 'John' and Minit = 'B' and Lname = 'Smith' and ssn = essn and pno = pnumber and project.dno = department.dnumber and Dept_Locations.dnumber = Department.dnumber and Plocation = Dlocation;

Q.9. Write SQL query to find the name of projects going on in Administration department.

SQL > Select pname from department, Dept_locations where dname = 'Administration and dno = dnumber;

Q.10. Write SQL query to find the dependent name of employees working on projects whose location is same to the location of administration department.

SQL > select dependent_name from project, department, employee, dept_locations, works_on where ssn = Dependent.essn = Works_on.essn and pno = pnumber and Department.dnumber and Dept_Locations.dnumber = Department.dnumber and Plocation = Dlocation and dname = 'Administration';

Experiment - 7

Aim: – Write SQL queries for sub queries, nested queries.

Q1. Make a list of project numbers for projects that involve an employee whose last name is 'Smith', either as a worker or as a manager of the department that controls the project.

```
SQL> SELECT DISTINCT Pnumber FROM PROJECT WHERE Pnumber
IN(SELECT Pnumber FROM PROJECT, DEPARTMENT, EMPLOYEE
WHERE Dnum= Dnumber AND Mgr_ssn = Ssn AND Lname = 'Smith')
OR Pnumber IN(SELECT Pno FROM WORKS_ON, EMPLOYEE WHERE
Essn = Ssn AND Lname = 'Smith');
```

Q2. Write SQL query to find the names of employees whose salary is greater than the salary of all the employees in department 5.

```
SQL > select fname from employee where salary >(select max
(salary) from employee where dno=5);
```

Q.3. Write SQL query to find the name of projects whose location is same to the location of research department.

```
SQL > SELECT E.Fname, E.Lname FROM EMPLOYEE AS E WHERE
E.Ssn IN (SELECT Essn FROM DEPENDENT AS D WHERE
E.Fname = D.Dependent_name AND E.Sex = D.Sex);
```

Q.4. Retrieve the name of each employee who has a dependent with the same first name and is the same sex as the employee.

```
SQL > SELECT E.Fname, E.Lname FROM EMPLOYEE AS E WHERE
E.Ssn IN( SELECT Essn FROM DEPENDENT AS D WHERE E.Fname
= D.Dependent_name AND E.Sex = D.Sex);
```

Q5. Retrieve the names of employees who have no dependents.

```
SQL> SELECT Fname, Lname FROM EMPLOYEE WHERE NOT EXISTS
(SELECT * FROM DEPENDENT WHERE Ssn = Essn);
```

Q6. List the names of managers who have at least one dependent.

SQL > SELECT Fname, Lname FROM EMPLOYEE WHERE EXISTS (SELECT * FROM DEPENDENT WHERE Ssn=Essn) AND EXISTS (SELECT * FROM DEPEN' DEPARTMENT WHERE Ssn= Mgr_ssn);

Q.7. Retrieve the Social Security numbers of all employees who work on project numbers 1, 2 or 3.

SQL >SELECT DISTINCT Essn FROM WORKS_ON WHERE Pno IN (1,2,3);

Q.9. Find the name of employees who are older than manager of Headquarters department.

SQL > SELECT Fname from Employee where bdate < (select bdate from department where ssn=mgr_ssn);

Q.10. Find the number of employees ~~where salary > (select m~~ is greater than minimum salary of research department

SQL > Select Fname from employee where salary > (select min (salary) from employee, department where dno = dnumber and dname = 'Research');

LAB - 8

Objective :- Write program of PL/SQL.

Q1. Write PL/SQL program to print Hello World.

```
DECLARE
-- variable declaration
message varchar2(20) := 'Hello, World!';
BEGIN
/*
* PL/SQL executable statement(s)
*/
dbms_output.put_line(message);
END;
```

Q2. Write PL/SQL program to create a procedure named greeting which on execution prints welcome message.

```
CREATE PROCEDURE greetings
AS
BEGIN
dbms_output.put_line('Hello World!');
END;
execute greetings;
```

Q3. Write a program to find Minimum of the two given numbers using procedure

```
DECLARE
a number;
b number;
c number;
PROCEDURE findmin(x IN number, y IN number, z OUT number)
IS BEGIN
IF x < y THEN
```

```
    z := x;
    ELSE
    z := y;
    END IF;
    END;
    BEGIN
    a := 23;
    b := 45;
    findMin(a,b,c);
    dbms_output.put_line('Minimum of (23,45): ' || c);
    END;
```

Q4. Write PL/SQL program to define variables and the value of these variables from existing databases then print the variables.

```
DECLARE
emp_id customers.id%type := 123456789;
emp_name customers.name%type;
emp_addr customers.address%type;
emp_sal customers.salary%type;
BEGIN
SELECT fname, address, salary INTO emp_name, emp_addr,
emp_sal
FROM employee
WHERE ssn = emp_id;
dbms_output.put_line
('Customer ' || emp_name || 'from' || emp_addr || 'earns' || emp_sal);
END;
```

LAB - 9

Objective :- Create VIEWS, CURSORS and TRIGGERS.

Q1. Create a view named emp1 and having attribute name and address derived from table Employee.

```
Create View Emp1 (name, Address)
As select Fname, Address from Employee.
```

Q2. Join the two tables Employee and Department and declare it a view and fire queries on it.

```
Create view Emp2
from Employee, Department where ssn= mgrssn;
```

Q3. Find the name of manager of research department using view created in query 2.

```
Select Fname From Emp2 where dname = research;
```

Q4. Create a trigger named Emp_Updated which prints the message table updated when some insertion or updation is performed on Employee table.

```
create trigger emp1 after update or insert on employee

begin
dbms_output.put_line('table updated');

end;
```

Q5. Create a cursor named Emp_cursor which prints name and address of employees working in department number 4 and then fetch the cursor and print the values pointed by EMP_cursor.

```
DECLARE Emp_id Employee.ssn%type;
Emp_name Employee.fname% type;
Emp_addr Employee.address% type;
```

```
CURSOR Emp_cursor is SELECT ssn, fname, address FROM employee;
BEGIN OPEN Emp_cursor;
LOOP FETCH Emp_cursor into Emp_id, Emp_name, Emp_addr;
dbms_output.put_line(Emp_id || ' ' || Emp_name || ' ' || Emp_addr);
EXIT WHEN Emp_cursor%notfound;
END LOOP;
update employee set address = 'XXYYZZ' where fname= 'James';
dbms_output.put_line ('Rows affected' || Emp_cursor%rowcount);
CLOSE Emp_cursor;
END;
```

LAB – 10

Objective :- Concepts for ROLL BACK, COMMIT & CHECK POINTS.

Q1. Create table Student and invert value and for each row create save points like sp1, sp2, sp3.

```
SQL > create table Student (RollNo, int, Name Varchar(20),
Address varchar(20), Age int, Sex char(5));
SQL > insert into student values (1, 'Arun', 'Lucknow', 21, 'male');
SQL > savepoint sp1;
```

Q2. Rollback the transaction upto savepoint 2.

```
SQL > Rollback to sp2;
```

Q3. Rollback upto savepoint 1.

```
SQL > Rollback to sp1;
```

Q4. Commit the transaction.

```
SQL > commit;
```