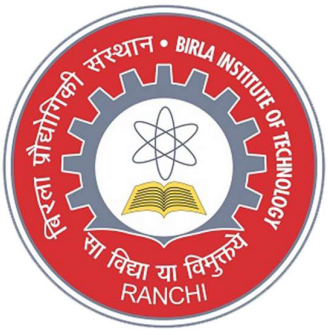
Birla Institute of Technology,Mesra,
Patna Campus



DBMS-LAB

Name-Shubham Sourabh

Roll-Btech/15044/18

Sec-CSE 6th

#Assignment

1.Create the employee database and it's corresponding tables using the schema given.

```
CREATE DATABASE DB;
use db;

DROP TABLE Employee;
DROP TABLE Department;
DROP TABLE Dependent;
DROP TABLE Project;
CREATE TABLE Employee(
    FNAME varchar(20),
    MINIT varchar(1),
    LNAME varchar(20),
    SSN int,
    BDATE varchar(15),
    ADDRESS varchar(50),
    SEX varchar(1),
    SALARY int,
    SUPERSSN int,
    DNO int
);
CREATE TABLE Department(
    DNAME varchar(30),
```

```
    DNumber int,
    MGRSSN int,
    SUPERSSN int,
    MGRSTARTDATE varchar(15)
);
CREATE TABLE Dependent(
    ESSN int,
    DEPENDENTNAME varchar(30),
    SEX varchar(1),
    BDATE varchar(15),
    RELATIONSHIP varchar(20)
);
CREATE TABLE Project(
    PNAME varchar(30) DEFAULT 'CSE',
    PNUMBER int DEFAULT 1,
    PLOCATION varchar(50) DEFAULT 'Patna',
    DNO int DEFAULT 0
);
```

2. Create a Teacher table from employee table with values of name, salary, birth-date and SSN.

```
CREATE TABLE Teachers AS SELECT FNAME, LNAME, BDATE,
SALARY, SSN FROM Employee;
show tables;
```

3. (a). Implementation of the Update command in employee table.

(b). Increase salaries of all male employees by 10% and female employees by 20%.

```
UPDATE Employee SET FNAME='ABC', LNAME='CDE' WHERE SUPERSSN=420;

UPDATE Employee SET BDATE='20 Oct 1972',SALARY=20000 WHERE SUPERSSN=422;


UPDATE Employee SET SALARY=40000  WHERE SEX='F' AND SALARY=35000;



UPDATE Employee SET salary= salary + (salary * 10 / 100) WHERE salary > 35000 AND SEX='M';

UPDATE Employee SET salary= salary + (salary * 20 / 100) WHERE salary <= 35000 AND SEX='F';
```

4. Create a table computer-science containing details of all employees working in this department and subsequently update the salaries of all employees working in this department by 50000.

```
CREATE TABLE CSE(     ENAME varchar(30),     SUPERSSN int, SALARY INT ,     BDATE varchar(50));
```

```
INSERT INTO CSE(SELECT FNAME,SUPERSSN,SALARY,BDATE
FROM Employee WHERE SUPERSSN IN (SELECT SUPERSSN
FROM Department WHERE DNAME='CSE'));
```

```
CREATE TABLE CSEDEPT SELECT
FNAME,SUPERSSN,SALARY,BDATE FROM Employee WHERE
SUPERSSN IN (SELECT SUPERSSN FROM Department WHERE
DNAME='CSE')
```

```
CREATE TABLE CSESALFIFTYGRTR SELECT
FNAME,SUPERSSN,SALARY,BDATE FROM CSEDEPT WHERE
SALARY>50000;
```

5. Implementation of the DELETE Command.

```
SELECT * FROM Employee;
```

```
DELETE FROM Employee WHERE SUPERSSN=420;
```

6. (a.) Display details of employees working in the management department

(b.) Implement the ROLLBACK concept.

(c.) Fetch and display the different projects available from the management department.

```
INSERT INTO Department VALUES('Management',2,2,425,'6 JAN');

INSERT INTO Department VALUES('Management',2,3,426,'7 JAN');

INSERT INTO Employee VALUES('Harry','J','Potter',425,'6 Jan 1996','London','M',80000,425,425);

INSERT INTO Employee VALUES('Ron','L','Whisley',426,'7 Jan 1996','London','M',76000,426,426);

SELECT FNAME,LNAME,SUPERSSN FROM Employee WHERE SUPERSSN IN (SELECT SUPERSSN FROM Department WHERE DNAME='Management');



DROP TABLE Project;


CREATE TABLE Project(
    PNAME varchar(30) DEFAULT 'CSE',
    PNUMBER int DEFAULT 1,
    PLOCATION varchar(50) DEFAULT 'Patna',
    DNO int DEFAULT 0
);

INSERT INTO Project VALUES('ML',27,'Patna',1);

INSERT INTO Project VALUES('DBMS',28,'Ranchi',1);

INSERT INTO Project VALUES('PM',27,'Patna',2);

INSERT INTO Project VALUES('BUSINESS COMM.',27,'Patna',2);
```

```
SELECT * FROM Project;


COMMIT;
START TRANSACTION;
INSERT INTO Project VALUES('FLT',27,'Patna',1);
SAVEPOINT A;
UPDATE Project SET PNAME='French' WHERE PNUMBER=30;
SAVEPOINT B;
UPDATE Project SET PNAME='COI' WHERE PNUMBER=29;
SAVEPOINT C;
SELECT * FROM Project;
ROLLBACK TO C;
SELECT * FROM Project;
ROLLBACK TO B;
SELECT * FROM Project;
ROLLBACK TO A;
SELECT * FROM Project;
COMMIT;
ROLLBACK TO C;


Select project FROM department WHERE DEPT=management;
```

7. Create a new user for the database and grant and revoke privileges to the former.

```
CREATE ROLE user1 WITH
    LOGIN
    SUPERUSER
    CREATEDB
    CREATEROLE
    INHERIT
    NOREPLICATION
    CONNECTION LIMIT -1;

ALTER DEFAULT PRIVILEGES
GRANT ALL ON TABLES TO user1;

ALTER DEFAULT PRIVILEGES
    REVOKE ALL ON TABLES FROM user1;

ALTER ROLE user1
    NOLOGIN
    NOSUPERUSER
    NOCREATEDB
    NOCREATEROLE;

DROP USER user1;
```

8. Implementation of the following queries:-

(a.) Write an sql query to fetch those employees whose address starts from P and work under computer science department.

(b.) Write an sql query to fetch all employees whose first name ends with I and work under management department.

(c.) Write an sql query to fetch all those employees who work on project other than p1.

```
SELECT * FROM employee

--q1

SELECT fname FROM Employee WHERE address LIKE 'P%' AND
department = 'CSE';

--q2

SELECT fname FROM Employee WHERE FNAME LIKE '%i' AND
department = 'MANAGEMENT';


SELECT * FROM department

SELECT * FROM project

--q3

SELECT * FROM project WHERE NOT pnumber='1';
```

9. (a.) Write an sql query to display the total salary of each employee adding the salary with variable value.

(b.) Write an sql query to fetch the employees whose name begins with any 2 characters followed by a text "hn" and ending with any sequence of characters.

```
DO $$
DECLARE
   myVar INT := 10000;
BEGIN
   UPDATE Employee SET Salary = (myVar) + Salary;
END $$;
SELECT * FROM employee
--q5
INSERT INTO Employee VALUES('JOHN','M','Carter',426,'7 Jan
1996','London','M',76000,426,426);
SELECT * FROM Employee WHERE FNAME LIKE '__HN%';
```

10. (a.) Write an sql query to fetch all the employee id which are present in employee table and arrange these employee ids in descending order

(b.) Write an sql query to fetch all the employee name whose name begins with any 2 characters followed by a text ‘hn’ and ending with sequence of characters and result is in descending order

(c.) Write an sql query to display the total salary of each employee adding the salary with variable value and result is grouped on the basis of this new salary

(d.) Write a query to display all the project details belonged to cse department along with employee name and employee ID in ascending order

(e.) Write a query to find the employee details whose permanent city is Patna

```
SELECT fname FROM Employee ORDER BY ssn desc;


SELECT * FROM Employee WHERE FNAME LIKE '__HN%'ORDER
BY fname desc;


DO $$
DECLARE
  Var INT := 10000;
BEGIN
SELECT fname AS NAME,salary, Salary = (Var) + Salary;
FROM employee GROUP BY salary;
END $$;
SELECT * FROM employee


SELECT * FROM employee
SELECT * FROM department
SELECT * FROM project


SELECT employee.fname , employee.ssn, project.name
FROM employee
INNER JOIN department ON employee.dno =
department.dnumber
INNER JOIN project ON employee.dno = project.dno WHERE
department.name = 'cse';


--q5
```

SELECT fname from employee WHERE address = Patna;

11.(a.) Write an sql query to fetch the employee id that are present in both tables. Table names are employee details and employee salary

b.) Write an sql query to fetch the employee id that are present in employee details but not in employee salary

c.) Write an sql query to fetch student records whose library fine is more than cost of the book issued by the student

d.) Write an sql query to fetch employee record whose salary is less than 10000 and belong to Patna

e.) Write an sql query to fetch employee record whose project cost is greater than 1 lakh

```
SELECT * FROM employee
CREATE TABLE Empsal(
    FNAME varchar(20),
    MINIT varchar(1),
    LNAME varchar(20),
    SSN int,
    Eid int,
    SEX varchar(1),
    SALARY int,
    DNO int
);
```

```
INSERT INTO Empsal
VALUES('hema','k','kislay',1,1,'M',80000,7);

INSERT INTO Empsal
VALUES('anish','k','kislay',2,2,'M',80000,7);

INSERT INTO Empsal
VALUES('shubho','k','kislay',4,4,'M',80000,7);


--q1

SELECT

   empsal.Fname,

   empsal.Eid

FROM

   empsal

INNER JOIN employee

   ON employee.ssn = empsal.eid;


--q2

SELECT * FROM empsal

SELECT

   employee.Fname,

   employee.ssn

FROM

   employee

INNER JOIN empsal

   ON NOT employee.ssn = empsal.eid;


--q3
```

```
CREATE TABLE student(NAME varchar(20), ID int, BOOK_ID
varchar(10), PRICE int, FINE int);

INSERT INTO student VALUES('hema',1,101,400,500);

INSERT INTO student VALUES('anish',2,102,400,300);

INSERT INTO student VALUES('shubho',3,103,400,700);


select * from student where fine > price;


--q4

SELECT fname,salary FROM Employee WHERE SALARY<50000
AND ADDRESS='London';


--q5


SELECT * FROM project

alter table project

add pcost int,ssn int;

INSERT INTO Project VALUES('try1',1,'Patna',1,100000,1);

INSERT INTO Project VALUES('try2',2,'Patna',7,40000,2);

INSERT INTO Project VALUES('try3',4,'Patna',1,500000,4);


SELECT
   *
FROM
   employee
INNER JOIN project
   ON  employee.ssn = project.ssn
```

```
where pcost>100000;
```

12. Implementation of the Division operator

```
create table COURSE_TAKEN(
Student_name VARCHAR(255),
Course VARCHAR(255)
);

create table COURSE_REQUIRED(
Course VARCHAR(255)
);

select * from COURSE_TAKEN

INSERT INTO COURSE_TAKEN VALUES('ROBERT','db');
INSERT INTO COURSE_TAKEN VALUES('ROBERT','plang');
INSERT INTO COURSE_TAKEN VALUES('david','db');
INSERT INTO COURSE_TAKEN VALUES('david','os');
INSERT INTO COURSE_TAKEN VALUES('hannah','plang');
INSERT INTO COURSE_TAKEN VALUES('hannah','ml');
INSERT INTO COURSE_TAKEN VALUES('tom','os');

select * from COURSE_REQUIRED

INSERT INTO COURSE_REQUIRED VALUES('db');
```

```
INSERT INTO COURSE_REQUIRED VALUES('PLANG');


CREATE TABLE AllStudents AS SELECT DISTINCT
Student_name FROM COURSE_TAKEN;


select * from AllStudents


CREATE table StudentsAndRequired AS SELECT
AllStudents.Student_name, COURSE_REQUIRED.Course FROM
AllStudents, COURSE_REQUIRED;

select * from StudentsAndRequired



CREATE table StudentsAndNotTaken AS SELECT * FROM
StudentsAndRequired WHERE NOT EXISTS (Select * FROM
COURSE_TAKEN WHERE StudentsAndRequired.Student_name
= COURSE_TAKEN.Student_name AND
StudentsAndRequired.Course = COURSE_TAKEN.Course);

select * from StudentsAndNotTaken



CREATE table CannotGraduate AS SELECT DISTINCT
Student_name FROM StudentsAndNotTaken;

select * from CannotGraduate;



CREATE Table CanGraduate AS SELECT * FROM AllStudents
WHERE NOT EXISTS (SELECT * FROM CannotGraduate WHERE
CannotGraduate.Student_name = AllStudents.Student_name);
```

```
select * from CanGraduate;


SELECT DISTINCT x.Student_Name FROM Course_Taken AS x
WHERE NOT EXISTS(SELECT * FROM Course_Required AS y
WHERE NOT EXISTS(SELECT * FROM Course_Taken AS z
WHERE z.Student_name = x.Student_name AND z.Course =
y.Course ));
```

13. (a.) Write a SQL statement to display all the information of all salesmen.

b.) Write a SQL statement to display a string "This is SQL Exercise, Practice and Solution"

c.) Write a SQL statement to display specific columns like name and commission for all the salesmen.

d.) Write a query to display the columns in a specific order like order date, salesman id, order number and purchase amount from for all the orders.

e.) Write a query which will retrieve the value of salesman id of all salesmen, getting orders from the customers in orders table without any repeats.

f.) Write a SQL statement to display names and city of salesman, who belongs to the city of Delhi.

g.) Write a SQL statement to display all the information for those customers with a grade of 200.

h.) Write a SQL query to display the order number followed by order date and the purchase amount for each order which will be delivered by the salesman who is holding the ID 5001.

i.) Solve above query using tables salesperson(salesman_id , name , city , commission) , order(ord_no , purch_amt,

ord_date , customer_id , salesman_id) and customer(customer_id , cust_name , city , grade , salesman_id)

```
create table salesman(eid int,name varchar(240),city varchar(240),commision int);


INSERT INTO salesman
VALUES(1,'Thomas','Birmingham',3000);

INSERT INTO salesman VALUES(2,'John','Birmingham',3000);

INSERT INTO salesman VALUES(3,'Arthur','London',3000);

INSERT INTO salesman VALUES(4,'Polly','Paris',3000);

INSERT INTO salesman VALUES(5,'Michael','Texas',3000);

--q1

SELECT * FROM salesman;

--q2

SELECT 'This is SQL Exercise, Practice and Solution';

--q3

SELECT name,commision FROM salesman;

--q4

create table orders( ono int, amt int ,odate date, customer_id int,eid int);


INSERT INTO orders VALUES(1,25000,'2021-07-18',1,1);

INSERT INTO orders VALUES(2,25000,'2021-07-16',2,2);

INSERT INTO orders VALUES(3,25000,'2021-07-14',3,3);

INSERT INTO orders VALUES(4,25000,'2021-07-11',4,4);
```

```
INSERT INTO orders VALUES(5,25000,'2021-07-10',5,5);


SELECT  * FROM orders ORDER BY odate;

SELECT  * FROM orders ORDER BY eid;

SELECT  * FROM orders ORDER BY amt;

--q5

select eid,name,city,commision from salesman where eid in(
select eid from orders);

--q6

SELECT name FROM salesman WHERE city='Paris';

--q7

create table customers( customer_id int, cname
varchar(100),city varchar(100),grade varchar(50),eid int);

INSERT INTO customers VALUES(1,'Billy
Kimber','Birmingham','196',1);

INSERT INTO customers VALUES(2,'Mr.Batten','Texas','200',2);

INSERT INTO customers VALUES(3,'Charles','London','196',3);

INSERT INTO customers
VALUES(4,'Changretta','Liverpool','200',4);

INSERT INTO customers VALUES(5,'ALfie
Solomons','Texas','196',5);


SELECT * FROM customers WHERE grade='200';


--q8

SELECT ono, odate, amt FROM orders WHERE eid=1;
```

14. (a.) Write an sql query to fetch 50% record from the employee info table

b.) Write a query to find the third highest salary from the employee position table

c.) Write a query to retrieve the last 3 records from the employee info table

d.) Write a query to find the nth highest salary from the table without using limit keyword

e.) Write a query to retrieve 2 minimum and maximum salaries from the employee position table

```
CREATE TABLE empinfo(
    eid int not null,
    FNAME varchar(90),
    LNAME varchar(90),
    DEPT varchar(90),
    project varchar(90),
    ADDRESS varchar(90),
    dob varchar(90),
    gender varchar(10)
);
```

```
INSERT INTO empinfo
VALUES(101,'tommy','shelby','Dev','p1','patna','20-05-2000','m');
```

```
INSERT INTO empinfo
VALUES(102,'arthur','shelby','Dev','p2','patna','20-11-
2000','m');

INSERT INTO empinfo
VALUES(101,'john','shelby','Mng','p3','patna','12-05-2000','m');

INSERT INTO empinfo
VALUES(101,'ada','shelby','Mng','p4','patna','12-05-2000','f');


CREATE TABLE emppos(
    eid int not null,
    emp_pos varchar(90),
    emp_join varchar(90),
    dofj varchar(90),
    salary int
);
select* from empinfo;
alter table emppos drop dofj;


INSERT INTO emppos VALUES(101,'sde','12-05-2000',10000);
INSERT INTO emppos VALUES(102,'sde','31-05-2000',20000);
INSERT INTO emppos VALUES(101,'gm','12-02-2000',40000);
INSERT INTO emppos VALUES(101,'cm','12-06-2000',60000);


--q1
select * from (select *, ntile(2) over(order by eid ) nt from
empinfo) as T where nt=1);


--q2
```

```
SELECT  Salary FROM emppos ORDER BY Salary DESC LIMIT 1
OFFSET 2;


--q3
SELECT  * FROM empinfo  LIMIT 3 OFFSET 2;


--q4
SELECT *
FROM emppos Emp1
WHERE (4) = (
SELECT COUNT( DISTINCT ( Emp2.salary ) )
FROM emppos Emp2
WHERE Emp2.salary >= Emp1.salary
)


--q5
select distinct salary from emppos order by salary desc limit 2;


select distinct salary from emppos order by salary asc limit 2;
```

___THANK YOU___