DBMS LAB WORK

Name: Satkar Gautam

Roll No: WRC078BCT034

1.(Exercise: retrieve the records from the table) EMPLOYEES (Employee_Id, First_Name, Last_Name,Email, Phone_Number, Hire_Date, Job_Id, Salary, Commission_Pct, Manager_Id, Department_Id)

1. create an employee's table with the following fields: (Emp_id, First_name, Last_name, Phone_No,Hire_date,Job_id,Emp_Salary,Comission_Pct,manager _id,Department_id)

```
CREATE TABLE employees ( Emp_id
        INT PRIMARY KEY,
        First_name VARCHAR(50) NOT NULL,
        Last_name VARCHAR(50) NOT NULL,
        Phone_No VARCHAR(10) NOT NULL,
        Hire_date DATE NOT NULL,
        Job_id VARCHAR(15) NOT NULL,
        Emp_Salary DOUBLE NOT NULL,
        Commission_Pct          DOUBLE,
        manager_id   INT,   Department_id
        INT)  ;
```

2. Insert five records into the table employees
(inserting values on every attribute so no need to mention attribute)

```
   INSERT INTO employees VALUES
(1, 'Ramesh', 'Shrestha', '9811122233', '2022-06-15', 'MANAGER', 95000.00, 0.04, 5, 12),
(2, 'Sita', 'Bhandari', '9822233344', '2021-09-30', 'DEVELOPER', 87000.00, 0.03, 4, 10),
(3, 'Bikash', 'Gurung', '9833344455', '2020-12-20', 'ANALYST', 78000.00, 0.025, 3, 8),
(4, 'Anita', 'Tamang', '9844455566', '2023-03-10', 'HR', 82000.00, 0.02, 4, 9),
(5, 'Prakash', 'Rai', '9855566677', '2024-01-05', 'CEO', 120000.00, 0.05, 6, 15);
```

3. Display the table Employees

```
SELECT * FROM employees;
```

4. Find out the employee id, names, salaries of all the employees

```
SELECT Emp_id,First_name,Last_name,Emp_Salary FROM employees;
```

5. Find the names of the employees who have a salary greater than or equal to 4800

```
SELECT First_name,Last_name,Emp_Salary FROM employees WHERE Emp_Salary>=4800;
```

6. List out the employees whose last name is ‘AUSTIN’

```
SELECT * FROM employees WHERE Last_name=’AUSTIN’;
```

7. Find the names of the employees who works in departments 60,70 and 80

```
SELECT First_name,Last_name FROM employees WHERE Department_id IN (60,70,80);
```

8. Display the unique Manager_Id from employees table

```
SELECT DISTINCT manager_id from employees;
```

2. (Exercise: update the records in the table) Create Client_master with the following fields (ClientNO,Name, Address, City, State, bal_due)

1. create a client master table with attributes

```
CREATE TABLE Client_Master (
        ClientNo VARCHAR(10) PRIMARY KEY,
        Name VARCHAR(100) NOT NULL,
        City VARCHAR(50) NOT NULL,
State VARCHAR(30) NOT NULL,
        bal_due DOUBLE);
```

2. insert five records into the Client_Master

```
INSERT INTO Client_Master VALUES
('C001', 'Santosh', 'Pokhara', 'Gandaki', '4000'),
('C002', 'Aarati', 'Kathmandu', 'Bagmati', '5000'),
('C003', 'Bikram', 'Biratnagar', 'Koshi', '3500'),
('C004', 'Sarita', 'Butwal', 'Lumbini', '6000'),
('C005', 'Pradeep', 'Dharan', 'Koshi', '4500');
```

3. Display Client Master Table

```
SELECT * FROM Client_Master;
```

4. Find the name of Clients whose balance_due >5000

```
SELECT Name FROM Client_Master WHERE bal_due >5000;
```

5. Change the bal_due of ClientNO “C123” to Rs. 5100

```
UPDATE Client_Master SET bal_due =5100 WHERE ClientNo = ‘C123’;
```

6. Change the name of Client_master to Client12

```
ALTER TABLE Client_Master RENAME TO Client12
```

7. Display the bal_due heading as “BALANCE” Client master table

```
SELECT ClientNo, Name, City, State, bal_due AS BALANCE FROM Client_Master;
```

3. Commands of Rollback and Commit : Create Teacher table with the following fields (Name, DeptNo,Date of joining, DeptName, Location, Salary)

1. Create Teacher table with the following fields (Id,Name, DeptNo, Date of joining, DeptName, Location, Salary)

```
CREATE TABLE Teacher (
        Id INT PRIMARY KEY,
        Name VARCHAR(100) NOT NULL,
        DeptNo INT NOT NULL,
        DateOfJoining DATE NOT NULL,
        DeptName VARCHAR(50) NOT NULL,
        Location VARCHAR(50) NOT NULL,
        Salary DOUBLE);
```

2. Insert five records

```
 INSERT INTO Teacher (Id, Name, DeptNo, DateOfJoining, DeptName, Location, Salary)
VALUES
(1, 'Ramesh Adhikari', 201, '2021-03-12', 'Electronics', 'Dharan', 72000.00),
(2, 'Sita Bhandari', 202, '2019-07-08', 'Commerce', 'Biratnagar', 75000.00),
(3, 'Bikash Tamang', 203, '2018-11-15', 'Mechanical', 'Hetauda', 78000.00),
(4, 'Anju Basnet', 204, '2020-06-22', 'Mathematics', 'Janakpur', 70000.00),
(5, 'Prakash Shrestha', 205, '2017-09-30', 'Computer Science', 'Chitwan', 82000.00);
```

3. Give Increment of 25% salary for Mathematics Department.

```
UPDATE Teacher SET Salary = Salary * 1.25 WHERE DeptName = ‘Mathematics’;
```

4. Perform Rollback command

```
UPDATE Teacher SET Salary = Salary * 1.25 WHERE DeptName = ‘Mathematics’;
```

(This can be also the better option)

```
START TRANSACTION;
UPDATE Teacher SET Salary = Salary * 1.25 WHERE DeptName = 'Mathematics';
ROLLBACK;
```

5. Give Increment of 15% salary for Commerce Department

```
UPDATE Teacher SET Salary = Salary * 1.25 WHERE DeptName = ‘Commerce’;
```

6. Perform commit command

COMMIT

4. (Exercise on the group by and order by clauses) Create Sales table with the following fields (SalesNo, Salesname, Branch, Salesamount, DOB)

1. Create a Sales Table with the following fields (Sales_No,Sales_Name,Branch,Sales_Amount,DOB)

```
CREATE TABLE Sales(
        Sales_No INT PRIMARY KEY,
        Sales_Name VARCHAR(100) NOT NULL,
        Branch VARCHAR(50) NOT NULL,
        Sales_Amount DOUBLE,
        DOB DATE NOT NULL);
```

2. Insert five records

```
INSERT INTO Sales (Sales_No, Sales_Name, Branch, Sales_Amount, DOB)
VALUES
(1, 'Ramesh Sharma', 'Dharan', 52000, '1995-07-18'),
(2, 'Kiran Bhandari', 'Bhaktapur', 63000, '1997-04-10'),
(3, 'Suman Rai', 'Hetauda', 77000, '1991-06-25'),
(4, 'Anil Magar', 'Janakpur', 56000, '1993-01-15'),
(5, 'Deepak Shrestha', 'Chitwan', 59000, '1989-11-30');
```

3. Calculate total salesamount in each branch

```
SELECT Branch, SUM(Sales_Amount) As Total_Sales_Amount FROM Sales GROUP BY
Branch;
```

4. Calculate average salesamount in each branch

```
SELECT Branch,AVG(Sales_Amount) As Total_Sales_Amount FROM Sales GROUP BY
Branch;
```

5. Display all the salesmen, DOB who are born in the month of December as day in character format i.e. 21-Dec-09

```
SELECT Sales_Name, DATE_FORMAT(DOB, '%d-%b-%y') AS DOB_Char FROM Sales
WHERE MONTH(DOB) = 12;
```

6. Display the name and DOB of salesman in alphabetical order of the month.

```
SELECT Sales_Name, DATE_FORMAT(DOB, '%d-%b-%y') AS DOB_Char FROM Sales
ORDER BY MONTH(DOB)
```

5. Create an Emp table with the following fields: (EmpNo, EmpName, Job,Basic, DA, HRA,PF, GrossPay,NetPay)

1. create an employee table with the following fields: (Emp_No,Emp_ Name, Designation, basic, DA, HRA, PF, Gross pay, Net pay)

```
CREATE TABLE employee( Emp_No
          INT PRIMARY KEY,
          Emp_Name VARCHAR(50), Designation
          VARCHAR(50),
          basic FLOAT,
          DA FLOAT,
          HRA FLOAT,
          PF FLOAT,
          Gross_pay FLOAT,
          Net_pay FLOAT);
```

2. Insert Five Records and calculate GrossPay and NetPay.

```
INSERT INTO employee (EmpNo, EmpName, Job, Basic, DA, HRA, PF)
VALUES
(1, 'Santosh Shrestha', 'Analyst', 51000, 7300, 10200, 5100),
(2, 'Kiran Magar', 'Consultant', 46000, 6700, 9200, 4600),
(3, 'Bikash Tamang', 'Supervisor', 49000, 6900, 9700, 4900),
(4, 'Anju Basnet', 'Project Manager', 53000, 7400, 10800, 5300),
(5, 'Rajendra Lama', 'Coordinator', 44000, 6200, 8900, 4400);
```

```
UPDATE employee SET Gross_Pay = Basic + DA + HRA, Net_Pay = Gross_Pay - PF;
```

3. Adding column to table and Updating Attributes DA

```
ALTER TABLE employee ADD AD2 FLOAT;
UPDATE employee SET DA2 = DA *1.10;
```

4. Adding column to table and Updating Attributes HRA

```
ALTER TABLE employee ADD HRA2 FLOAT;
UPDATE employee SET HRA2 = HRA * 1.15;
```

5. Adding column to table and Updating Attributes PF

```
ALTER TABLE employee ADD PF2 FLOAT;
UPDATE employee SET PF2 = PF * 0.95;
```

6. Adding column to table and Updating Attributes Gross Pay

```
ALTER TABLE employee ADD Goss_pay2 FLOAT;
UPDATE employee SET Gross_pay2 = Basic + DA2 + HRA2;
```

7. Adding column to table and Updating Attributes Net Pay

```
ALTER TABLE employee ADD Net_pay2 FLOAT;
UPDATE employee SET Net_pay2 = Gross_pay2 - PF2;
```

8. Display the employee table

```
SELECT * FROM employee
```

9. Display the employees whose Basic is lowest in each department.

```
SELECT Emp_Name, Basic, Designation FROM employee WHERE Basic =
(SELECT MIN(Basic) FROM employee WHERE Designation = employee.Designation);
```

10. If NetPay is less than Rs. 10,000 add Rs. 1200 as special allowance

```
UPDATE Employee SET Net_pay = Net_pay + 1200 WHERE Net_Pay < 10000;
```

11. Display the employees whose GrossPay lies between 10,000 & 20,000

```
SELECT Emp_Name, Gross_pay FROM employee WHERE Gross_pay BETWEEN 10000 AND
20000;
```

12. Display all the employees who earn maximum salary

```
SELECT Emp_Name, Gross_pay FROM employee WHERE Gross_pay = ( SELECT
MAX(Gross_pay) FROM employee;
```

6. Employee Database an Enterprise wishes to maintain a database to automate its operations. Enterprise is divided into certain departments and each department consists of employees. The following two tables describes the automation schemas Dept (deptno, dname, loc) Emp (empno, ename, job, mgr, hiredate, sal, comm, deptno)

1. Create Dept table: Dept (deptno, dname, loc)

```
CREATE TABLE Dept(
        deptno INT PRIMARY KEY, dname
        VARCHAR(100) NOT NULL, loc
        VARCHAR(100) NOT NULL);
```

2. Create Dept table: Emp (empno, ename, job, mgr, hiredate, sal, comm, deptno)

```
CREATE TABLE Emp(          empno INT
PRIMARY KEY,          ename
VARCHAR(255) NOT NULL,            job
VARCHAR(255) NOT NULL,            mgr
INT NOT NULL,         hiredate DATE NOT
NULL,            sal DECIMAL(10,2) NOT
NULL,            comm DECIMAL(10,2),
deptno INT NOT NULL,
        FOREIGN KEY(deptno) REFERENCES Dept(deptno));
```

3. Insert data int Dept and Emp tables

```
INSERT INTO Dept (deptno, dname, loc)
VALUES
(10, 'Administration', 'Kathmandu'),
(20, 'Accounting', 'Pokhara'),
(30, 'Software Development', 'Lalitpur'),
(40, 'Public Relations', 'Biratnagar'),
(50, 'Customer Support', 'Butwal');
```

```
INSERT INTO Emp (empno, ename, job, mgr, hiredate, sal, comm, deptno)
VALUES
(301, 'Rajesh Sharma', 'Manager', 0, '2015-06-15', 95000.00, 5000.00, 10),
(302, 'Sita Thapa', 'Software Engineer', 301, '2016-03-12', 75000.00, NULL, 20),
(303, 'Bikash Gurung', 'Data Analyst', 301, '2018-08-25', 85000.00, 6000.00, 30),
(304, 'Nirjana Koirala', 'HR Specialist', 302, '2019-11-09', 58000.00, 4000.00, 40),
(305, 'Manoj Rai', 'Customer Service', 304, '2020-07-17', 47000.00, 3000.00, 50);
```

4. Update the employee salary by 15%, whose experience is greater than 30 years

```
UPDATE Emp SET sal=sal*1.15 WHERE YEAR(hiredate)< YEAR(CURRENT_DATE)-30 ;
```

5. Delete the employees, who completed 30 years of service.

```
DELETE FROM Emp WHERE YEAR(hiredate) <= YEAR(CURRENT_DATE)-30;
```

6. Display the manager who is having maximum number of employees working under him?

SELECT mgr,COUNT(*) AS Num_Employees FROM Emp GROUP BY mgr ORDER BY Num_Employees DESC LIMIT 1;

7. Create a view, which contain employee names and their manager

CREATE VIEW EmployeeManagerView AS SELECT e1.name AS EmployeeName, e2.ename AS ManagerName From Emp e1 LEFT JOIN Emp e2 ON e1.mgr = e2.mgr;

7. Using Employee Database above perform the following queries

a) Determine the names of employee, who earn more than their managers.

SELECT e1.ename AS EmployeeName FROM Emp e1 JOIN Emp e2 ON e1.mgr = e2.empno WHERE e1.sal > e2.sal ;

b) Determine the names of employees, who take highest salary in their departments.

SELECT e.ename, e.sal, d.dname FROM Emp e JOIN Dept d ON e.deptno = d.deptno WHERE e.sal = (SELECT MAX(sal) FROM Emp WHERE deptno = e.deptno);

c) Determine the employees, who are located at the same place.

SELECT e1.ename AS Employee1, e2.ename AS Employee2 FROM Emp e1 JOIN Emp e2 ON e1.deptno = e2.deptno AND e1.empno <> e2.empno;
d) Determine the employees, whose total salary is like the minimum Salary of any department.

SELECT e.ename FROM Emp e WHERE e.sal = (SELECT MIN(sal) FROM Emp WHERE deptno = e.deptno);

e) Determine the department which does not contain any employees

SELECT d.dname FROM Dept d LEFT JOIN Emp e ON d.deptno = e.deptno WHERE e.empno IS NULL;

8. Using the tables “DEPARTMENTS” and “EMPLOYEES” above perform the following queries

a) Display the employee details, departments that the departments are same in both the emp and dept.

```
SELECT e.empno, e.deptno, d,dname FROM emp e JOIN dept d ON e.deptno = d.deptno;
```

b) Display the employee name and Department name by implementing a left outer join.

```
SELECT e.ename AS EmployeeName, d.dname AS DepartmentName FROM Emp e   LEFT
JOIN Dept d ON e.deptno = d.deptno;
```

c) Display the employee name and Department name by implementing a right outer join.

```
SELECT e.ename AS EmployeeName, d.dname AS DepartmentName FROM Emp e   RIGHT
JOIN Dept d ON e.deptno = d.deptno;
```

d) Display the details of those who draw the salary greater than the average salary.

```
SELECT empno, ename, sal  FROM Emp WHERE sal > (SELECT AVG(sal) FROM Emp);
```