**Experiment - 1**

**1. Create a table EMPLOYEE with following schema:**

***(Emp\_no, E\_name, E\_address, E\_ph\_no, Dept\_no, Dept\_name,Job\_id , Salary)***

Create table EMPLOYEE

CREATE TABLE EMPLOYEE (

Emp\_no INT,

E\_name VARCHAR(50),

E\_address VARCHAR(100),

E\_ph\_no VARCHAR(15),

Dept\_no INT,

Dept\_name VARCHAR(50),

Job\_id CHAR(5),

Salary DECIMAL(10, 2)

);

**2. Add a new column; HIREDATE to the existing relation.**

-- Add a new column HIREDATE

ALTER TABLE EMPLOYEE

ADD HIREDATE DATE;

**3. Change the datatype of JOB\_ID from char to varchar2.**

-- Change datatype of JOB\_ID from char to varchar

ALTER TABLE EMPLOYEE

ALTER COLUMN Job\_id VARCHAR(5);

**4. Change the name of column/field Emp\_no to E\_no.**

-- Change column name Emp\_no to E\_no

EXEC sp\_rename 'EMPLOYEE.Emp\_no', 'E\_no', 'COLUMN';

**5. Modify the column width of the job field of emp table**

-- Modify column width of the job field

ALTER TABLE EMPLOYEE

ALTER COLUMN Job\_id VARCHAR(10);

**Experiment - 2**

**1. Insert aleast 5 rows in the table.**

INSERT INTO EMP (Emp\_no, Emp\_name, Department, City, Email\_id)

VALUES (1, 'Alice', 'D10', 'New York', 'alice@example.com'),

(2, 'Bob', 'D20', 'Los Angeles', 'bob@example.com'),

(3, 'Charlie', 'D10', 'Chicago', 'charlie@example.com'),

(4, 'David', 'MECH', 'Houston', 'david@example.com'),

(5, 'Eve', 'SALES', 'Miami', 'eve@example.com');

**2. Display all the information of EMP table.**

SELECT \* FROM EMP;

**3. Display the record of each employee who works in department D10.**

SELECT \* FROM EMP WHERE Department = 'D10';

**4. Update the city of Emp\_no-12 with current city as Nagpur.**

UPDATE EMP SET City = 'Nagpur' WHERE Emp\_no = 12;

**5. Display the details of Employee who works in department MECH.**

SELECT \* FROM EMP WHERE Department = 'MECH';

**6. Delete the email\_id of employee James.**

UPDATE EMP SET Email\_id = NULL WHERE Emp\_name = 'James';

**7. Display the complete record of employees working in SALES Department.**

SELECT \* FROM EMP WHERE Department = 'SALES';

**Experiment No: 3**

**1. List the E\_no, E\_name, Salary of all employees working for MANAGER**

SELECT E\_no, E\_name, Salary

FROM employees

WHERE Designation = 'MANAGER';

**2. Display all the details of the employee whose salary is more than the Sal of any IT PROFF..**

SELECT \*

FROM employees

WHERE Salary > (SELECT MAX(Sal) FROM employees WHERE Designation = 'IT PROFF.');

**3. List the employees in the ascending order of Designations of those joined after 1981.**

SELECT \*

FROM employees

WHERE Join\_Date > '1981-01-01'

ORDER BY Designation ASC;

**4. List the employees along with their Experience and Daily Salary.**

SELECT E\_no, E\_name, DATEDIFF(YEAR, Join\_Date, GETDATE()) AS Experience, Salary/30 AS Daily\_Salary

FROM employees;

**5. List the employees who are either ‘CLERK’ or ‘ANALYST’ .**

SELECT \*

FROM employees

WHERE Designation IN ('CLERK', 'ANALYST');

**6. List the employees who joined on 1-MAY-81, 3-DEC-81, 17-DEC-81,19-JAN-80**

SELECT \*

FROM employees

WHERE Join\_Date IN ('1981-05-01', '1981-12-03', '1981-12-17', '1980-01-19');

**7. List the employees who are working for the Deptno 10 or20**

SELECT \*

FROM employees

WHERE Deptno IN (10, 20);

**8. List the Enames those are starting with ‘S’ .**

SELECT E\_name

FROM employees

WHERE E\_name LIKE 'S%';

**9. Dislay the name as well as the first five characters of name(s) starting with ‘H’**

SELECT E\_name, LEFT(E\_name, 5) AS Short\_Name

FROM employees

WHERE E\_name LIKE 'H%';

**10. List all the emps except ‘PRESIDENT’ & ‘MGR” in asc order of Salaries.**

SELECT \*

FROM employees

WHERE Designation NOT IN ('PRESIDENT', 'MGR')

ORDER BY Salary ASC;

**Experiment No: 4**

**1. Display all the dept numbers available with the dept and emp tables avoiding duplicates.**

SELECT deptno

FROM dept

UNION

SELECT deptno

FROM emp;

**2. Display all the dept numbers available with the dept and emp tables**

SELECT DISTINCT deptno

FROM dept

JOIN emp ON dept.deptno = emp.deptno;

**3. Display all the dept numbers available in emp and not in dept tables and vice versa.**

SELECT emp.deptno

FROM emp

WHERE emp.deptno NOT IN (SELECT deptno FROM dept)

UNION

SELECT dept.deptno

FROM dept

WHERE dept.deptno NOT IN (SELECT deptno FROM emp);

**Experiment No: 5**

**Sailors (sid, sname, rating, age)**

**Boats (bid, bname, color)**

**Reserves (sid, bid, day(date))**

**1. Find all information of sailors who have reserved boat number 101.**

SELECT \*

FROM Sailors

WHERE sid IN (SELECT sid FROM Reserves WHERE bid = 101);

**2. Find the name of boat reserved by Bob.**

SELECT bname

FROM Boats

WHERE bid IN (SELECT bid FROM Reserves WHERE sid = (SELECT sid FROM Sailors WHERE sname = 'Bob'));

**3. Find the names of sailors who have reserved a red boat, and list in the order of age.**

SELECT s.sname

FROM Sailors s

JOIN Reserves r ON s.sid = r.sid

JOIN Boats b ON r.bid = b.bid

WHERE b.color = 'red'

ORDER BY s.age;

**4. Find the names of sailors who have reserved at least one boat.**

SELECT s.sname

FROM Sailors s

JOIN Reserves r ON s.sid = r.sid;

**5. Find the ids and names of sailors who have reserved two different boats on the same**

**day.**

SELECT s.sid, s.sname

FROM Sailors s

JOIN Reserves r ON s.sid = r.sid

GROUP BY s.sid, s.sname

HAVING COUNT(DISTINCT r.bid) = 2;

**6. Find the ids of sailors who have reserved a red boat or a green boat**

SELECT DISTINCT s.sid

FROM Sailors s

JOIN Reserves r ON s.sid = r.sid

JOIN Boats b ON r.bid = b.bid

WHERE b.color = 'red' OR b.color = 'green';

**7. Find the name and the age of the youngest sailor.**

SELECT TOP 1 s.sname, s.age

FROM Sailors s

ORDER BY s.age;

**8. Count the number of different sailor names.**

SELECT COUNT(DISTINCT s.sname) AS num\_sailors

FROM Sailors s;

**9. Find the average age of sailors for each rating level.**

SELECT rating, AVG(age) AS avg\_age

FROM Sailors

GROUP BY rating;

**10. Find the average age of sailors for each rating level that has at least two sailors**

SELECT rating, AVG(age) AS avg\_age

FROM Sailors

GROUP BY rating

HAVING COUNT(sid) >= 2;

**Experiment No: 6**

-- Create Employee table

CREATE TABLE Employee (

EmployeeID INT PRIMARY KEY,

EmployeeName VARCHAR(50),

JobCategory VARCHAR(50),

Salary DECIMAL(10, 2),

DepartmentID INT,

ManagerID INT

);

**1. Display total salary spent for each job category.**

SELECT JobCategory, SUM(Salary) AS TotalSalary

FROM Employee

GROUP BY JobCategory;

**2. Display lowest paid employee details under each manager.**

SELECT ManagerID, MIN(Salary) AS LowestSalary

FROM Employee

GROUP BY ManagerID;

**3. Display number of employees working in each department and their department name.**

SELECT DepartmentID, COUNT(EmployeeID) AS NumberOfEmployees

FROM Employee

GROUP BY DepartmentID;

**4. Display the details of employees sorting the salary in increasing order.**

SELECT \*

FROM Employee

ORDER BY Salary;

**5. Show the record of employee earning salary greater than 16000 in each department.**

SELECT \*

FROM Employee

WHERE Salary > 16000

ORDER BY DepartmentID;

**Experiment No: 7**

**Consider the following schema:**

**Sailors (sid, sname, rating, age)**

**Boats (bid, bname, color)**

**Reserves (sid, bid, day(date))**

**Write subquery statement for the following queries.**

**1. Find all information of sailors who have reserved boat number 101.**

SELECT \*

FROM Sailors

WHERE sid IN (SELECT sid FROM Reserves WHERE bid = 101);

**2. Find the name of boat reserved by Bob.**

SELECT bname

FROM Boats

WHERE bid IN (SELECT bid FROM Reserves WHERE sid = (SELECT sid FROM Sailors WHERE sname = 'Bob'));

**3. Find the names of sailors who have reserved a red boat, and list in the order of age.**

SELECT sname

FROM Sailors

WHERE sid IN (SELECT sid FROM Reserves WHERE bid IN (SELECT bid FROM Boats WHERE color = 'red'))

ORDER BY age;

**4. Find the names of sailors who have reserved at least one boat.**

SELECT sname

FROM Sailors

WHERE sid IN (SELECT sid FROM Reserves);

**5. Find the ids and names of sailors who have reserved two different boats on the same**

**day.**

SELECT s.sid, s.sname

FROM Sailors s

WHERE EXISTS (SELECT \* FROM Reserves r1 WHERE r1.sid = s.sid

AND EXISTS (SELECT \* FROM Reserves r2 WHERE r2.sid = s.sid AND r1.day = r2.day AND r1.bid <> r2.bid));

**6. Find the ids of sailors who have reserved a red boat or a green boat.**

SELECT sid

FROM Reserves

WHERE bid IN (SELECT bid FROM Boats WHERE color IN ('red', 'green'));

**7. Find the name and the age of the youngest sailor.**

SELECT TOP 1 sname, age

FROM Sailors

ORDER BY age;

**8. Count the number of different sailor names.**

SELECT COUNT(DISTINCT sname)

FROM Sailors;

**9. Find the average age of sailors for each rating level.**

SELECT rating, AVG(age) AS avg\_age

FROM Sailors

GROUP BY rating;

**10. Find the average age of sailors for each rating level that has at least two sailors.**

SELECT rating, AVG(age) AS avg\_age

FROM Sailors

GROUP BY rating

HAVING COUNT(sid) >= 2;

**Experiment No: 8**

**1. Create a table called EMP with the following structure.**

**Name Type**

**EMPNO NUMBER (6)**

**ENAME VARCHAR2 (20)**

**JOB VARCHAR2 (10)**

**DEPTNO NUMBER (3)**

**SAL NUMBER (7,2)**

**Allow NULL for all columns except ename and job.**

-- Create table EMP--

CREATE TABLE EMP (

EMP\_NO (INT) PRIMARY KEY,

E\_NAME VARCHAR(20) NOT NULL,

JOB VARCHAR(10) NOT NULL,

DEPT\_NO INT,

SAL DECIMAL(7,2),

CONSTRAINT CHK\_EMPNO CHECK (EMPNO > 100),

CONSTRAINT UNQ\_DEPTNO UNIQUE (DEPTNO),

CONSTRAINT PK\_EMPNO PRIMARY KEY (EMPNO)

);

**2. Add constraints to check, while entering the empno value (i.e) empno > 100.**

INSERT INTO EMP (EMPNO, ENAME, JOB, DEPTNO, SAL) VALUES (101, 'John Doe', 'Manager', 10, 5000.00);

INSERT INTO EMP (EMPNO, ENAME, JOB, DEPTNO, SAL) VALUES (102, 'Jane Smith', 'Analyst', 20, 4000.00);

**3. Define the field DEPTNO as unique.**

CONSTRAINT UNQ\_DEPTNO UNIQUE (DEPTNO),

**4. Create a primary key constraint for the table(EMPNO).**

CONSTRAINT PK\_EMPNO PRIMARY KEY (EMPNO)

**5. Write queries to implement and practice constraints.**

SELECT \* FROM EMP;

SELECT \* FROM EMP WHERE DEPTNO = 10;

**Experiment No: 9**

**1. Write a query to implement the save point.**

SAVE TRANSACTION SavePoint1;

**2. Write a query to implement the rollback.**

ROLLBACK TRANSACTION SavePoint1;

**3. Write a query to implement the commit.**

COMMIT TRANSACTION;

**Experiment No: 10**

**1. Create user and implement the following commands on relation (Emp and Dept).**

CREATE USER user\_name FOR LOGIN login\_name;

GRANT SELECT, INSERT, UPDATE, DELETE ON Emp TO user\_name;

GRANT SELECT, INSERT, UPDATE, DELETE ON Dept TO user\_name;

**2. Develop a query to grant all privileges of employees table into departments table.**

GRANT ALL ON Emp TO Dept;

**3. Develop a query to grant some privileges of employees table into departments table.**

GRANT SELECT, INSERT ON Emp TO Dept;

**4. Develop a query to revoke all privileges of employees table from departments table.**

GRANT SELECT, INSERT ON Emp TO Dept;

**5. Develop a query to revoke some privileges of employees table from departments table.**

REVOKE ALL ON Emp TO Dept;