**DML Queries:**

1. **Insert a new department named "Marketing" with dept\_id 4 and location "San Francisco".**
2. **Insert a new employee with emp\_id 6, emp\_name "Laura Anderson", dept\_id 1, emp\_position "Developer", and emp\_salary 4200.00.**
3. **Update the salary of employee with emp\_id 2 to 4500.00.**
4. **Delete the project with project\_id 3 from the Project table.**
5. **Insert a new entry in the Employee\_Project table to assign employee with emp\_id 3 to project with project\_id 1.**
6. **Update the manager\_name of employee with emp\_id 3 to "Michelle Roberts" in the Manager table.**
7. **Delete all employees from the Sales department (dept\_id 2) from the Employee table.**
8. **Insert a new salary record for employee with emp\_id 4, set the salary as 3800.00, and the effective\_date as the current date.**
9. **Update the salary of all employees in the IT department (dept\_id 1) by adding a 10% increment to their current salaries.**
10. **Delete all salary history records for employees with emp\_id 5 from the SalaryHistory table.**

**DDL Queries/Questions:**

1. **Create a table called "Customers" with columns cust\_id (INT), cust\_name (VARCHAR), and city (VARCHAR).**
2. **Add a new column "phone" with datatype VARCHAR(15) to the Employee table.**
3. **Create a foreign key constraint on the Employee table to enforce referential integrity with the Department table.**
4. **Drop the Manager table from the database.**
5. **Create an index on the emp\_name column in the Employee table for faster retrieval.**
6. **Rename the column "emp\_position" in the Employee table to "job\_title".**
7. **Add a primary key constraint to the Project table on the project\_id column.**
8. **Alter the data type of the salary column in the SalaryHistory table to DECIMAL(12, 2).**
9. **Create a view named "EmployeeDetails" that displays the employee name, department name, and project name for each employee.**
10. **Create a stored procedure named "GetEmployeeCountByDepartment" that accepts a department ID as input and returns the count of employees in that department.**

**Join Queries/Questions:**

1. **Retrieve the employee name, department name, and salary for all employees.**
2. **Retrieve the employee names and project names for all employees assigned to projects.**
3. **Retrieve the employee names and manager names for all employees who have a manager.**
4. **Retrieve the employee names, department names, and project names for all employees in the Sales department.**
5. **Retrieve the employee names and their respective salary history records.**
6. **Retrieve the employee names, project names, and start dates for all employees assigned to projects in the IT department.**
7. **Retrieve the employee names and project names for all employees who are not assigned to any project.**
8. **Retrieve the employee names, department names, and average salary for each department.**
9. **Retrieve the employee names and project names for all employees who are assigned to more than one project.**
10. **Retrieve the employee names and their respective manager names for all employees in the IT department.**

DML Queries:

Insert a new department named "Marketing" with dept\_id 4 and location "San Francisco":

INSERT INTO Department (dept\_id, dept\_name, location)

VALUES (4, 'Marketing', 'San Francisco');

Insert a new employee with emp\_id 6, emp\_name "Laura Anderson", dept\_id 1, emp\_position "Developer", and emp\_salary 4200.00:

INSERT INTO Employee (emp\_id, emp\_name, dept\_id, emp\_position, emp\_salary)

VALUES (6, 'Laura Anderson', 1, 'Developer', 4200.00);

Update the salary of employee with emp\_id 2 to 4500.00:

UPDATE Employee

SET emp\_salary = 4500.00

WHERE emp\_id = 2;

Delete the project with project\_id 3 from the Project table:

DELETE FROM Project

WHERE project\_id = 3;

Insert a new entry in the Employee\_Project table to assign employee with emp\_id 3 to project with project\_id 1:

INSERT INTO Employee\_Project (emp\_id, project\_id)

VALUES (3, 1);

Update the manager\_name of employee with emp\_id 3 to "Michelle Roberts" in the Manager table:

UPDATE Manager

SET manager\_name = 'Michelle Roberts'

WHERE emp\_id = 3;

Delete all employees from the Sales department (dept\_id 2) from the Employee table:

DELETE FROM Employee

WHERE dept\_id = 2;

Insert a new salary record for employee with emp\_id 4, set the salary as 3800.00, and the effective\_date as the current date:

INSERT INTO SalaryHistory (emp\_id, salary, effective\_date)

VALUES (4, 3800.00, CURDATE());

Update the salary of all employees in the IT department (dept\_id 1) by adding a 10% increment to their current salaries:

UPDATE Employee

SET emp\_salary = emp\_salary \* 1.1

WHERE dept\_id = 1;

Delete all salary history records for employees with emp\_id 5 from the SalaryHistory table:

DELETE FROM SalaryHistory

WHERE emp\_id = 5;

**DDL Queries/Questions:**

Create a table called "Customers" with columns cust\_id (INT), cust\_name (VARCHAR), and city (VARCHAR):

CREATE TABLE Customers (

cust\_id INT PRIMARY KEY,

cust\_name VARCHAR(50),

city VARCHAR(50)

);

Add a new column "phone" with datatype VARCHAR(15) to the Employee table:

ALTER TABLE Employee

ADD phone VARCHAR(15);

Create a foreign key constraint on the Employee table to enforce referential integrity with the Department table:

ALTER TABLE Employee

ADD CONSTRAINT fk\_department

FOREIGN KEY (dept\_id) REFERENCES Department(dept\_id);

Drop the Manager table from the database:

DROP TABLE Manager;

Create an index on the emp\_name column in the Employee table for faster retrieval:

CREATE INDEX idx\_emp\_name ON Employee(emp\_name);

Rename the column "emp\_position" in the Employee table to "job\_title":

ALTER TABLE Employee

RENAME COLUMN emp\_position TO job\_title;

Add a primary key constraint to the Project table on the project\_id column:

ALTER TABLE Project

ADD CONSTRAINT pk\_project\_id PRIMARY KEY (project\_id);

Alter the data type of the salary column in the SalaryHistory table to DECIMAL(12, 2):

ALTER TABLE SalaryHistory

ALTER COLUMN salary TYPE DECIMAL(12, 2);

Create a view named "EmployeeDetails" that displays the employee name, department name, and project name for each employee:

CREATE VIEW EmployeeDetails AS

SELECT emp.emp\_name, dept.dept\_name, proj.project\_name

FROM Employee emp

JOIN Department dept ON emp.dept\_id = dept.dept\_id

JOIN Employee\_Project ep ON emp.emp\_id = ep.emp\_id

JOIN Project proj ON ep.project\_id = proj.project\_id;

Create a stored procedure named "GetEmployeeCountByDepartment" that accepts a department ID as input and returns the count of employees in that department:

CREATE PROCEDURE GetEmployeeCountByDepartment(IN dept\_id INT, OUT emp\_count INT)

BEGIN

SELECT COUNT(\*) INTO emp\_count

FROM Employee

WHERE dept\_id = dept\_id;

END;

**Join Queries/Questions:**

Retrieve the employee name, department name, and salary for all employees:

SELECT emp.emp\_name, dept.dept\_name, emp.emp\_salary

FROM Employee emp

JOIN Department dept ON emp.dept\_id = dept.dept\_id;

Retrieve the employee names and project names for all employees assigned to projects:

SELECT emp.emp\_name, proj.project\_name

FROM Employee emp

JOIN Employee\_Project ep ON emp.emp\_id = ep.emp\_id

JOIN Project proj ON ep.project\_id = proj.project\_id;

Retrieve the employee names and manager names for all employees who have a manager```

SELECT emp.emp\_name, mgr.manager\_name

FROM Employee emp

JOIN Manager mgr ON emp.emp\_id = mgr.emp\_id;

Retrieve the employee names, department names, and project names for all employees in the Sales department:

SELECT emp.emp\_name, dept.dept\_name, proj.project\_name

FROM Employee emp

JOIN Department dept ON emp.dept\_id = dept.dept\_id

LEFT JOIN Employee\_Project ep ON emp.emp\_id = ep.emp\_id

LEFT JOIN Project proj ON ep.project\_id = proj.project\_id

WHERE dept.dept\_name = 'Sales';

Retrieve the employee names and their respective salary history records:

SELECT emp.emp\_name, sh.salary, sh.effective\_date

FROM Employee emp

JOIN SalaryHistory sh ON emp.emp\_id = sh.emp\_id;

Retrieve the employee names, project names, and start dates for all employees assigned to projects in the IT department:

SELECT emp.emp\_name, proj.project\_name, proj.start\_date

FROM Employee emp

JOIN Employee\_Project ep ON emp.emp\_id = ep.emp\_id

JOIN Project proj ON ep.project\_id = proj.project\_id

JOIN Department dept ON emp.dept\_id = dept.dept\_id

WHERE dept.dept\_name = 'IT';

Retrieve the employee names and project names for all employees who are not assigned to any project:

SELECT emp.emp\_name, proj.project\_name

FROM Employee emp

LEFT JOIN Employee\_Project ep ON emp.emp\_id = ep.emp\_id

LEFT JOIN Project proj ON ep.project\_id = proj.project\_id

WHERE ep.project\_id IS NULL;

Retrieve the employee names, department names, and average salary for each department:

SELECT emp.emp\_name, dept.dept\_name, AVG(emp.emp\_salary) AS avg\_salary

FROM Employee emp

JOIN Department dept ON emp.dept\_id = dept.dept\_id

GROUP BY emp.emp\_name, dept.dept\_name;

Retrieve the employee names and project names for all employees who are assigned to more than one project:

SELECT emp.emp\_name, proj.project\_name

FROM Employee emp

JOIN Employee\_Project ep ON emp.emp\_id = ep.emp\_id

JOIN Project proj ON ep.project\_id = proj.project\_id

GROUP BY emp.emp\_name

HAVING COUNT(ep.project\_id) > 1;

Retrieve the employee names and their respective manager names for all employees in the IT department:

SELECT emp.emp\_name, mgr.manager\_name

FROM Employee emp

JOIN Manager mgr ON emp.emp\_id = mgr.emp\_id

JOIN Department dept ON emp.dept\_id = dept.dept\_id

WHERE dept.dept\_name = 'IT';