



## Experiment 3

**Student Name :**Naresh Kumar

**Branch:** CSE

**Semester:** 5<sup>th</sup>

**Subject Name:** ADBMS

**UID:** 23BCS13655

**Section/Group:** KRG\_2B

**Date of Performance:** 18/08/25

**Subject Code:** 23CSP-333

### 1. Aim:

To design and implement SQL queries for creating tables, inserting data, and retrieving meaningful information using relational concepts.

- To apply aggregate functions, joins, subqueries, and set operations for solving database problems.

#### ◆ Part A – Easy Level:

- To create a table for storing employee IDs and insert sample data.
- To identify and retrieve the maximum employee ID that does not have duplicates.

#### ◆ Part B – Medium Level:

- To create department and employee tables with a foreign key relationship.
- To retrieve the employee(s) having the highest salary in each department using joins and subqueries.

#### ◆ Part C – Hard Level:

- To create two tables containing employee details with salaries.
- To combine the tables and retrieve the minimum salary for each employee using grouping and aggregate functions.

### 2. Objective:

- ✓ To understand the use of **GROUP BY** and **aggregate functions** for filtering data.
- ✓ To apply **joins and subqueries** for department-wise salary analysis.
- ✓ To implement **foreign key relationships** for relational database design.
- ✓ To use **UNION ALL and grouping** for analyzing data across multiple tables.
- ✓ To strengthen SQL query writing skills for handling duplicates, aggregation, and joins.



### 3. ADBMS script and output:

#### EASY-LEVEL PROBLEM

```
CREATE TABLE Employeee (
```

```
    EmpID INT ,
```

```
);
```

```
INSERT INTO Employeee (EmpID) VALUES
```

```
(2),
```

```
(4 ),
```

```
(4),
```

```
(6),
```

```
(6),
```

```
(7),
```

```
(8),
```

```
(8);
```

```
Select Max(EmpID) as [Maximum ID] from (Select EmpID from Employeee Group by EmpID  
having Count(*) < 2)as Subquery;
```

#### MEDIUM LEVEL PROBLEM:

```
CREATE TABLE departmentt (
```

```
    id INT PRIMARY KEY,
```

```
    dept_name VARCHAR(50)
```



# DEPARTMENT OF COMPUTER SCIENCE & ENGINEERING

Discover. Learn. Empower.

);

CREATE TABLE employeee (

id INT,

name VARCHAR(50),

salary INT,

department\_id INT,

FOREIGN KEY (department\_id) REFERENCES departmentt(id)

);

INSERT INTO departmentt (id, dept\_name) VALUES

(1, 'IT'),

(2, 'SALES');

INSERT INTO employeee (id, name, salary, department\_id) VALUES

(1, 'JOE', 70000, 1),

(2, 'JIM', 90000, 1),

(3, 'HENRY', 80000, 2),

(4, 'SAM', 60000, 2),

(5, 'MAX', 90000, 1);

Select d.dept\_name,e.name,e.salary from departmentt as d Join employeee as e on d.id =  
e.department\_id where e.salary in (

Select max(e2.salary) from employeee as e2 where e2.department\_id = e.department\_id);

## HARD LEVEL PROBLEM

Create table A1 (ID int , Ename varchar(50), Salary int);

Create Table B1(ID int, Ename varchar(50), Salary int );

Insert into A1 values(1,'AA',1000);

Insert into A1 values(2,'BB',300);

Insert into B1 values(2,'BB',400);



# DEPARTMENT OF COMPUTER SCIENCE & ENGINEERING

Discover. Learn. Empower.

Insert into B1 values(3,'CC',100);

Select ID, EName, Min(Salary) as Min\_Salary from

(Select \* from A1 Union All Select\* from B1) as combined Group by Ename,ID;

## OUTPUTS:

Results		Messages	
	Maximum ID		
1	7		

Figure 1: Easy Level Problem

	dept_name	name	salary
1	SALES	HENRY	80000
2	IT	MAX	90000
3	IT	JIM	90000

Figure 2: Medium level Problem



# DEPARTMENT OF COMPUTER SCIENCE & ENGINEERING

Discover. Learn. Empower.

	ID	EName	Min_Salary
1	1	AA	1000
2	2	BB	300
3	3	CC	100

Figure 3: Hard Level Problem