Experiment 3

Student Name: Ishan   **UID:** 23BCS12656
Branch: CSE   **Section/Group:** KRG_2B
Semester: 5th   **Date of Performance:** 19/08/25
Subject Name: ADBMS   **Subject Code:** 23CSP-333

1. Aim:

The purpose of this task is to design and implement SQL queries for creating relational tables, inserting records, and retrieving useful information. It emphasizes the use of aggregate functions, joins, subqueries, and set operations to handle different database scenarios.

Part A – Easy Level
- Create a table to store employee IDs and insert sample records.
- Write a query to display the maximum employee ID that appears only once (without duplicates).

Part B – Medium Level
- Create two tables: one for departments and one for employees, ensuring a foreign key relationship between them.
- Retrieve the employee(s) with the highest salary in each department using a combination of joins and subqueries.

Part C – Hard Level
- Create two tables containing employee details along with their salary information.
- Merge both tables using UNION ALL and write a query to find the minimum salary for each employee by applying grouping and aggregate functions.

Objectives
- To practice the use of GROUP BY and aggregate functions for filtering and summarizing data.
- To apply joins and subqueries for department-wise salary comparisons.
- To understand the implementation of foreign key relationships in relational database design.
- To use UNION ALL with grouping for analyzing combined records across multiple tables.
- To strengthen SQL query writing skills by working with duplicates, aggregations, and joins in different problem levels.

2. ADBMS script and output:

EASY-LEVEL PROBLEM

```
CREATE TABLE EmpData (
    EmpCode INT
);

INSERT INTO EmpData (EmpCode) VALUES
(2),
(4),
(4),
(6),
(6),
(7),
(8),
(8);

SELECT MAX(EmpCode) AS `Maximum_Code`
FROM (
    SELECT EmpCode
    FROM EmpData
    GROUP BY EmpCode
    HAVING COUNT(*) = 1
) AS T1;
```

MEDIUM LEVEL PROBLEM:

```
CREATE TABLE DeptInfo (
    DeptID INT PRIMARY KEY,
    DeptTitle VARCHAR(50)
);

CREATE TABLE StaffInfo (
    StaffID INT,
    FullName VARCHAR(50),
    Pay INT,
    DeptRef INT,
    FOREIGN KEY (DeptRef) REFERENCES DeptInfo(DeptID)
);

INSERT INTO DeptInfo (DeptID, DeptTitle) VALUES
(1, 'IT'),
```

(2, 'SALES');

INSERT INTO StaffInfo (StaffID, FullName, Pay, DeptRef) VALUES
(1, 'JOE', 70000, 1),
(2, 'JIM', 90000, 1),
(3, 'HENRY', 80000, 2),
(4, 'SAM', 60000, 2),
(5, 'MAX', 90000, 1);

SELECT d.DeptTitle, s.FullName, s.Pay
FROM DeptInfo d
JOIN StaffInfo s
   ON d.DeptID = s.DeptRef
WHERE s.Pay IN (
   SELECT MAX(x.Pay)
   FROM StaffInfo x
   WHERE x.DeptRef = s.DeptRef
);

HARD LEVEL PROBLEM

CREATE TABLE TableOne (

   PID INT,

   PersonName VARCHAR(50),

   Income INT

);

CREATE TABLE TableTwo (

   PID INT,

   PersonName VARCHAR(50),

   Income INT

);

INSERT INTO TableOne VALUES (1, 'AA', 1000);

INSERT INTO TableOne VALUES (2, 'BB', 300);

INSERT INTO TableTwo VALUES (2, 'BB', 400);

INSERT INTO TableTwo VALUES (3, 'CC', 100);

WITH AllData AS (

```
    SELECT PID, PersonName, Income FROM TableOne

    UNION ALL

    SELECT PID, PersonName, Income FROM TableTwo

)

SELECT

   PID,

   PersonName,

   MIN(Income) AS LowestIncome

FROM AllData

GROUP BY PID, PersonName;
```

OUTPUTS:

Output:

```
+---------------+
| Maximum_Code  |
+---------------+
|             7 |
+---------------+
```

Figure 1: Easy Level Problem

Output:

```
+-----------+----------+-------+
| DeptTitle | FullName | Pay   |
+-----------+----------+-------+
| IT        | JIM      | 90000 |
| SALES     | HENRY    | 80000 |
| IT        | MAX      | 90000 |
+-----------+----------+-------+
```

Figure 2: Medium level Problem

Output:

```
+------+------------+--------------+
| PID  | PersonName | LowestIncome |
+------+------------+--------------+
|    1 | AA         |         1000 |
|    2 | BB         |          300 |
|    3 | CC         |          100 |
+------+------------+--------------+
```

Figure 3: Hard Level Problem