University Institute of Engineering

Department of Computer Science & Engineering

EXPERIMENT : 3

NAME : Sayan Pradhan UID: 23BCS10878

BRANCH : BE-CSE SECTION/GROUP : KRG_3B

SEMESTER : 5TH SUBJECT CODE : 23CSP-333

SUBJECT NAME : ADBMS

1. Aim Of The Practical :

Max Value without Duplicates [EASY]

- Create a table of Employee IDs.
- Insert sample IDs (with duplicates).
- Write a query to return the maximum EmpID excluding duplicate values using subqueries.

Department Salary Champions [MEDIUM]

- Create `dept` and `employee` tables with a relationship.
- Insert sample department and employee data.
- Use subqueries to find the employee(s) with the highest salary in each department.
- If multiple employees share the max salary in a department, include all.

Merging Employee Histories: Who Earned Least? [HARD]

- Create two legacy tables (TableA and TableB).
- Insert sample records (some overlapping).
- Merge both tables and find the minimum salary per employee using subqueries.

2. Tools Used: SQL Server Management Studio

3. Code:

```
-              -easy question
          CREATE TABLE EMPLOYEE (
              EMPID INT
             );

         INSERT INTO EMPLOYEE VALUES (1),(2),(3),(2),(4),(6),(6),(7),(7);

          SELECT * FROM EMPLOYEE;

         SELECT MAX(EMPID) AS MaxUniqueEmpID
        FROM EMPLOYEE
              WHERE EMPID NOT IN (
           SELECT EMPID
          FROM EMPLOYEE
        GROUP BY EMPID
          HAVING COUNT(EMPID) > 1
             );


           --medium  question

REATE TABLE dept (
id INT PRIMARY KEY,
Dept_Name VARCHAR(50) NOT NULL
);

CREATE TABLE employee (
id INT PRIMARY KEY,
EmpName VARCHAR(50),
Salary INT,
Dept_Id INT FOREIGN KEY REFERENCES dept(id)
);

INSERT INTO dept VALUES (1, 'IT'), (2, 'SALES');

INSERT INTO employee VALUES
(1, 'JOE', 70000, 1),
(2, 'JIM', 90000, 1),
(3, 'HENRY', 80000, 2),
(4, 'SAM', 60000, 2),
(5, 'MAX', 90000, 1);

-- Get top earners in each department
SELECT D.Dept_Name, E.EmpName, E.Salary
FROM employee AS E
INNER JOIN dept AS D
ON E.Dept_Id = D.id
WHERE E.Salary IN (
SELECT MAX(E2.Salary)
FROM employee AS E2
WHERE E2.Dept_Id = E.Dept_Id
);


-- hard question

CREATE TABLE TableA (
Empid INT,
```

```
Ename VARCHAR(50),
Salary INT
);

CREATE TABLE TableB (
Empid INT,
Ename VARCHAR(50),
Salary INT
);

INSERT INTO TableA VALUES (1, 'AA', 1000), (2, 'BB', 300);
INSERT INTO TableB VALUES (2, 'BB', 400), (3, 'CC', 100);

-- Find each employee with minimum salary across both tables
SELECT Empid, Ename, MIN(Salary) AS LowestSalary
FROM (
SELECT Empid, Ename, Salary FROM TableA
UNION ALL
SELECT Empid, Ename, Salary FROM TableB
) AS Combined
GROUP BY Empid, Ename;
```

4. Output :

[EASY]

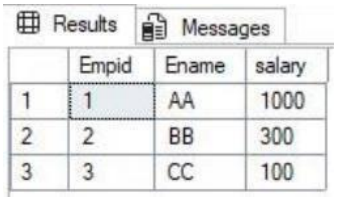
	empid
1	4

[MEDIUM]

Results | Messages

	department_id	salary	id
1	2	80000	2
2	1	90000	1
3	1	90000	1

[HARD]



Results | Messages

	Empid	Ename	salary
1	1	AA	1000
2	2	BB	300
3	3	CC	100

5. Learning Outcomes:

- Learn to create and define relational database tables using the CREATE TABLE command, along with understanding common data types such as `INT` and VARCHAR.
- Build practical skills in setting up primary keys to ensure each record can be uniquely identified.
- Understand how to define and enforce foreign key constraints to preserve data consistency between linked tables (e.g., Books linked to Authors).
- Gain the ability to perform INNER `JOIN` operations to merge records from multiple tables using a shared key (such as author_id).
- Learn how to structure normalized relational schemas with foreign key relationships for real-world examples like departments and courses. • Become comfortable inserting several rows into related tables using the INSERT INTO statement.
- Master the use of subqueries alongside GROUP BY and HAVING to summarize and filter aggregated results.
- Apply query logic to select data from a parent table based on conditions derived from aggregated results in a related child table.