21.7.25

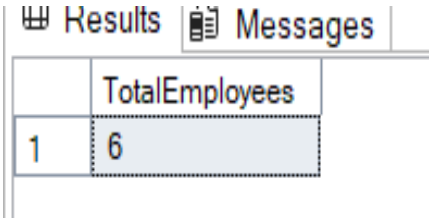
CODING CHALLENGE – 1 SQL

AGGREGATE FUNCTIONS

1. COUNT – Total number of employees

Definition: COUNT() returns the number of rows.

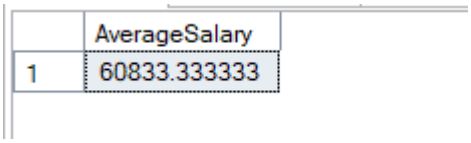
Query: SELECT COUNT(*) AS TotalEmployees FROM Employees;



2. AVG – Average salary of employees

Definition: AVG() calculates the average value of a numeric column.

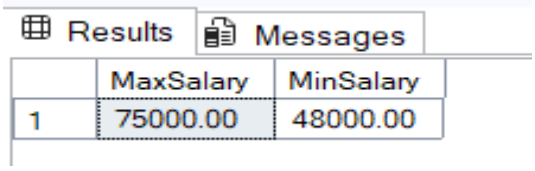
Query : SELECT AVG(Salary) AS AverageSalary FROM Employees;



3. MAX / MIN – Find highest and lowest salary

Definition: MAX() gets the highest value, MIN() gets the lowest.

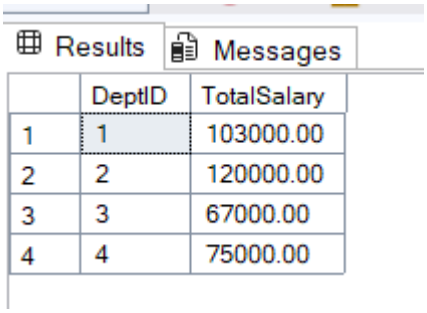
Query : SELECT MAX(Salary) AS MaxSalary, MIN(Salary) AS MinSalary FROM Employees;



4. SUM – Total salary by department

Definition: SUM() adds up all the salary values in a group.

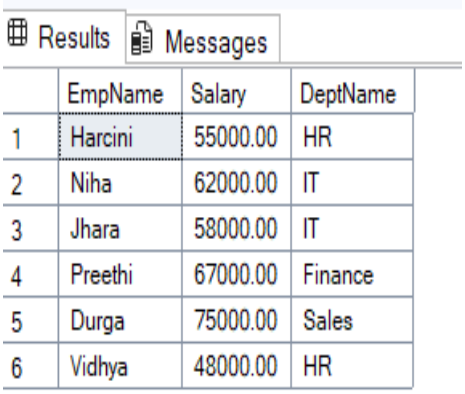
Query: SELECT DeptID, SUM(Salary) AS TotalSalary FROM Employees GROUP BY DeptID;



JOINS

1. INNER JOIN

Show employees with their department names.



	EmpName	Salary	DeptName
1	Harcini	55000.00	HR
2	Niha	62000.00	IT
3	Jhara	58000.00	IT
4	Preethi	67000.00	Finance
5	Durga	75000.00	Sales
6	Vidhya	48000.00	HR

Definition:
INNER JOIN returns only rows that have matching values in both tables.

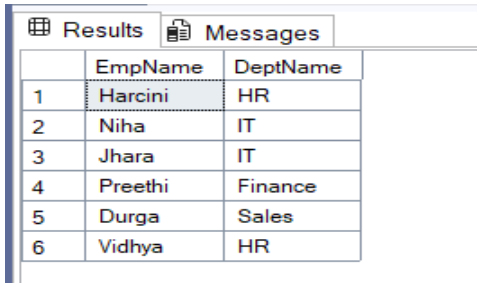
Query:

```
SELECT e.EmpName, e.Salary, d.DeptName
FROM Employees e
INNER JOIN Departments d ON e.DeptID = d.DeptID;
```

2. LEFT JOIN (LEFT OUTER JOIN)

Question: Show all employees, even those who don't belong to a department.

Definition:
LEFT JOIN returns all rows from the left table (Employees) and the matched rows from the right table (Departments). If no match, NULLs are returned from the right side.



	EmpName	DeptName
1	Harcini	HR
2	Niha	IT
3	Jhara	IT
4	Preethi	Finance
5	Durga	Sales
6	Vidhya	HR

Query:

```
SELECT e.EmpName, d.DeptName
FROM Employees e
LEFT JOIN Departments d ON e.DeptID = d.DeptID;
```

3. RIGHT JOIN (RIGHT OUTER JOIN)

Question: Show all departments, even if they have no employees.

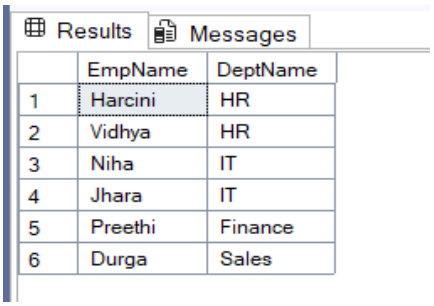
Definition:
RIGHT JOIN returns all rows from the right table (Departments), and the matched rows from the left table (Employees). NULLs are returned if no match.

Query:

```
SELECT e.EmpName, d.DeptName
```

FROM Employees e

RIGHT JOIN Departments d ON e.DeptID = d.DeptID;

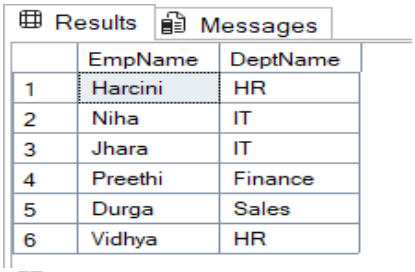


	EmpName	DeptName
1	Harcini	HR
2	Vidhya	HR
3	Niha	IT
4	Jhara	IT
5	Preethi	Finance
6	Durga	Sales

4. FULL OUTER JOIN

Question: Show all employees and departments, even if they don't match.

Definition:
FULL OUTER JOIN returns all rows from both tables. If there's no match, NULLs are used for missing values.



	EmpName	DeptName
1	Harcini	HR
2	Niha	IT
3	Jhara	IT
4	Preethi	Finance
5	Durga	Sales
6	Vidhya	HR

Query:

SELECT e.EmpName, d.DeptName

FROM Employees e

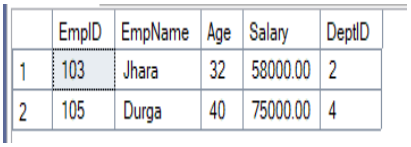
FULL OUTER JOIN Departments d ON e.DeptID = d.DeptID;

FILTERING DATA

1. **Employees older than 30**

Definition: WHERE filters rows based on a condition.

Query: SELECT * FROM Employees WHERE Age > 30;

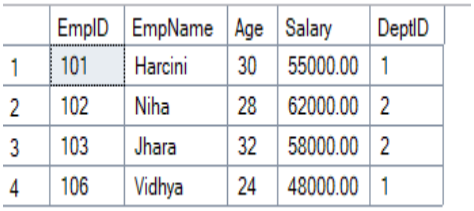


	EmpID	EmpName	Age	Salary	DeptID
1	103	Jhara	32	58000.00	2
2	105	Durga	40	75000.00	4

2. **Employees from HR or IT**

Definition: IN checks if a value matches any value in a list.

Query: SELECT * FROM Employees WHERE DeptID IN (1, 2);

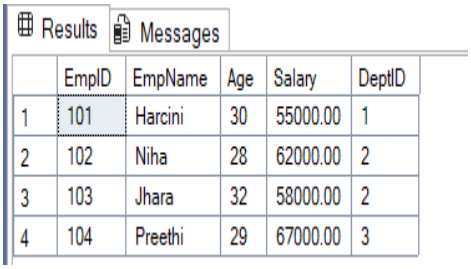


	EmpID	EmpName	Age	Salary	DeptID
1	101	Harcini	30	55000.00	1
2	102	Niha	28	62000.00	2
3	103	Jhara	32	58000.00	2
4	106	Vidhya	24	48000.00	1

3. **Salary between 50,000 and 70,000**

Definition: BETWEEN filters values in a given range.

Query: SELECT * FROM Employees WHERE Salary BETWEEN 50000 AND 70000;



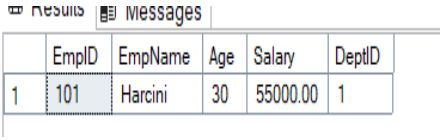
	EmpID	EmpName	Age	Salary	DeptID
1	101	Harcini	30	55000.00	1
2	102	Niha	28	62000.00	2
3	103	Jhara	32	58000.00	2
4	104	Preethi	29	67000.00	3

4. **Names starting with ‘H’**

Definition: LIKE 'H%' filters strings that start with 'H'.

Query : SELECT * FROM Employees

WHERE EmpName LIKE 'H%';



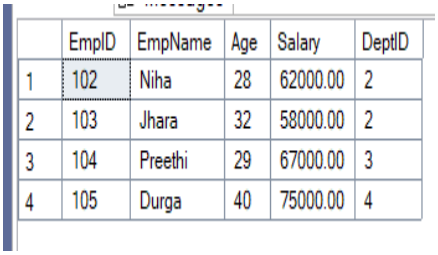
	EmpID	EmpName	Age	Salary	DeptID
1	101	Harcini	30	55000.00	1

5. **Age > 25 and Salary > 55000**

Definition: AND allows multiple conditions to be applied together.

Query : SELECT * FROM Employees

WHERE Age > 25 AND Salary > 55000;



	EmpID	EmpName	Age	Salary	DeptID
1	102	Niha	28	62000.00	2
2	103	Jhara	32	58000.00	2
3	104	Preethi	29	67000.00	3
4	105	Durga	40	75000.00	4

GROUP BY and HAVING

1. **Average salary by department where avg > 60000**

Definition: GROUP BY groups rows, HAVING filters those groups.

Query:

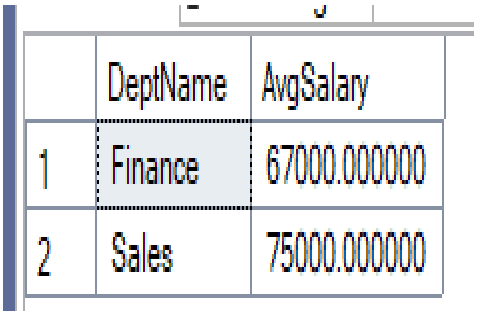
SELECT d.DeptName, AVG(e.Salary) AS AvgSalary

FROM Employees e

JOIN Departments d ON e.DeptID = d.DeptID

GROUP BY d.DeptName

HAVING AVG(e.Salary) > 60000;



	DeptName	AvgSalary
1	Finance	67000.000000
2	Sales	75000.000000