

## SQL Coding Challenge [\(21.07.2025\)](#)

### SQL Query:

```
CREATE DATABASE CompanyData;
```

```
USE CompanyData;
```

#### -- Departments Table

```
CREATE TABLE Departments (
```

```
    DeptID INT PRIMARY KEY,
```

```
    DeptName VARCHAR(50)
```

```
);
```

#### -- Employees Table

```
CREATE TABLE Employees (
```

```
    EmpID INT PRIMARY KEY,
```

```
    EmpName VARCHAR(50),
```

```
    Age INT,
```

```
    Salary DECIMAL(10, 2),
```

```
    DeptID INT FOREIGN KEY REFERENCES Departments(DeptID)
```

```
);
```

#### -- Insert Departments

```
INSERT INTO Departments VALUES
```

```
(1, 'HR'),
```

```
(2, 'IT'),
```

```
(3, 'Finance'),
```

```
(4, 'Sales'),
```

```
(5, 'Testing');
```

#### -- Insert Employees

```
INSERT INTO Employees VALUES
```

```
(101, 'Abi', 30, 55000, 1),
```

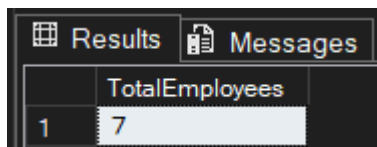
```
(102, 'Akalya', 28, 62000, 2),
```

```
(103, 'Renu', 32, 58000, 5),
```

(104, 'SelvaPriya', 29, 67000, 3),  
(105, 'Priya', 40, 75000, 4),  
(106, 'Mugunthan', 26, 50000, 5),  
(107, 'Tharani', 38, 80000, 4);

#### -- Total number of employees

```
SELECT COUNT(*) AS TotalEmployees FROM Employees;
```



The screenshot shows a SQL query result window with two tabs: 'Results' and 'Messages'. The 'Results' tab is active, displaying a single row with the column header 'TotalEmployees' and the value '7'.

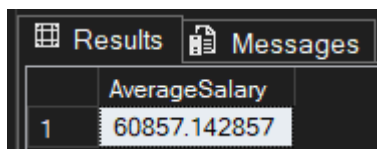
TotalEmployees
7

#### Explanation:

- Calculates the total number of employee records in the Employees table.
- COUNT(\*) is an aggregate function that counts all records; FROM specifies the table source.

#### -- Average salary

```
SELECT AVG(Salary) AS AverageSalary FROM Employees;
```



The screenshot shows a SQL query result window with two tabs: 'Results' and 'Messages'. The 'Results' tab is active, displaying a single row with the column header 'AverageSalary' and the value '60857.142857'.

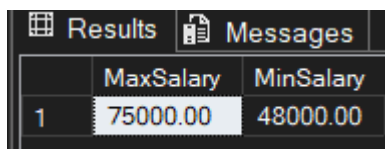
AverageSalary
60857.142857

#### Explanation:

- Helps calculate the average salary of all employees.
- AVG(Salary) computes the mean value of the Salary column.

#### -- Highest and Lowest Salary

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



The screenshot shows a SQL query result window with two tabs: 'Results' and 'Messages'. The 'Results' tab is active, displaying a single row with two columns: 'MaxSalary' and 'MinSalary'. The values are '75000.00' and '48000.00' respectively.

MaxSalary	MinSalary
75000.00	48000.00

#### Explanation:

- Used to get the highest and lowest salaries among employees.
- MAX() returns the highest value; MIN() returns the lowest value in the selected column.

### -- Total Salary by Department

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

Results		Messages
	DeptID	TotalSalary
1	1	258000.00
2	2	395000.00
3	3	343000.00
4	4	364000.00
5	5	266000.00

#### Explanation:

- Calculates total salary expense per department.
- SUM(Salary) adds all salaries; GROUP BY groups data based on department ID.

### -- INNER JOIN (Employees + Departments)

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

Results		Messages	
	EmpName	Salary	DeptName
1	Abi	55000.00	HR
2	Akalya	62000.00	IT
3	Renu	58000.00	Testing
4	SelvaPriya	67000.00	Finance
5	Priya	75000.00	Sales
6	Mugunthan	48000.00	HR
7	Thaara	61000.00	IT

#### Explanation:

- Used to get employee details along with their department names.
- JOIN ... ON connects rows from two tables where DeptID matches.

### -- LEFT JOIN

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

FROM Employees e

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

	EmpName	DeptName
1	Abi	HR
2	Akalya	IT
3	Renu	Testing
4	SelvaPriya	Finance
5	Priya	Sales
6	Mugunthan	HR
7	Thaara	IT

#### Explanation:

- Displays all employees, even if some don't belong to any department.
- LEFT JOIN returns all records from the left table and matched records from the right table.

#### -- RIGHT JOIN

SELECT e.EmpName, d.DeptName

FROM Employees e

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

	EmpName	DeptName
1	Abi	HR
2	Mugunthan	HR
3	Akalya	IT
4	Thaara	IT
5	SelvaPriya	Finance
6	Priya	Sales
7	Renu	Testing

#### Explanation:

- Lists all departments, even those with no assigned employees.
- RIGHT JOIN returns all records from the right table and matched records from the left table.

#### --Employees older than 30

SELECT \* FROM Employees WHERE Age > 30;

	EmpID	EmpName	Age	Salary	DeptID
1	103	Renu	32	58000.00	5
2	105	Priya	40	75000.00	4

#### Explanation:

- Filters and shows employees whose age is greater than 30.
- WHERE clause filters rows; > checks if age is above the specified number.

#### -- Employees from HR or IT

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

	EmpID	EmpName	Age	Salary	DeptID
1	101	Abi	30	55000.00	1
2	102	Akalya	28	62000.00	2
3	106	Mugunthan	24	48000.00	1
4	107	Thaara	27	61000.00	2

#### Explanation:

- Used to retrieve employees from specific departments like HR or IT.
- IN operator checks if DeptID matches any value in the given list.

#### -- Employees with salary between 50000 and 70000

SELECT \* FROM Employees WHERE Salary BETWEEN 50000 AND 70000;

	EmpID	EmpName	Age	Salary	DeptID
1	101	Abi	30	55000.00	1
2	102	Akalya	28	62000.00	2
3	103	Renu	32	58000.00	5
4	104	SelvaPriya	29	67000.00	3
5	107	Thaara	27	61000.00	2

#### Explanation:

- Filters employees whose salaries fall within a given range.
- BETWEEN is used to specify the lower and upper bounds for filtering.

### -- Names starting with 'A'

```
SELECT * FROM Employees WHERE EmpName LIKE 'A%';
```

Results		Messages			
	EmpID	EmpName	Age	Salary	DeptID
1	101	Abi	30	55000.00	1
2	102	Akalya	28	62000.00	2

#### Explanation:

- Finds all employees whose names begin with the letter "A".
- LIKE 'A%' matches names starting with 'A'; % is a wildcard.

### -- Multiple conditions: Age > 25 AND Salary > 55000

```
SELECT * FROM Employees WHERE Age > 25 AND Salary > 55000;
```

Results		Messages			
	EmpID	EmpName	Age	Salary	DeptID
1	102	Akalya	28	62000.00	2
2	103	Renu	32	58000.00	5
3	104	SelvaPriya	29	67000.00	3
4	105	Priya	40	75000.00	4
5	107	Thaara	27	61000.00	2

#### Explanation:

- Filters records based on two conditions: age and salary.
- AND combines both conditions and returns only records meeting both.