

# Sql code Case study

```
-- Employee Table
```

```
create table employee (
    employeeid int primary key,
    name varchar(50),
    departmentid int,
    salary int,
    hiredate date
);
```

```
drop table employee
```

```
-- Department Table
```

```
create table department (
    departmentid int primary key,
    departmentname varchar(50)
);
```

```
-- Performance Table
```

```
create table performance (
    performanceid int primary key,
    employeeid int,
    reviewdate date,
    score int,
    foreign key (employeeid) references employee(employeeid)
);
```

```
-- Departments
```

```
insert into department values
```

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

```
(2, 'Engineering'),
```

```
(3, 'Marketing');
```

```
-- Employees
```

```
insert into employee values
```

```
(1, 'Arjun', 2, 75000, '2023-02-15'),
```

```
(2, 'Meena', 1, 60000, '2024-07-01'),
```

```
(3, 'Ravi', 2, 85000, '2024-03-10'),
```

```
(4, 'Sneha', 3, 72000, '2022-09-05'),
```

```
(5, 'Kumar', 1, 90000, '2024-08-20');
```

```
-- Performance
```

```
insert into performance values
```

```
(101, 1, '2025-01-10', 88),
```

```
(102, 2, '2025-02-14', 92),
```

```
(103, 3, '2025-01-30', 95),
```

```
(104, 4, '2024-11-20', 78),
```

```
(105, 5, '2025-01-15', 85);
```

```
select * from employee
```

```
select * from department
```

```
select * from performance
```

The screenshot shows the SSMS interface with the following details:

- Query Window:** The main window displays a T-SQL script named "Practicetask1". It includes data insertions into three tables: employee, department, and performance.
- employee Table Data:**

	employeeid	name	departmentid	salary	hiredate
1	1	Arjun	2	75000	2023-02-15
2	2	Meena	1	60000	2024-07-01
3	3	Ravi	2	85000	2024-03-10
4	4	Sneha	3	72000	2022-09-05
5	5	Kumar	1	90000	2024-08-20
- department Table Data:**

	departmentid	departmentname
1	1	HR
2	2	Engineering
3	3	Marketing
- performance Table Data:**

	performanceid	employeeid	reviewdate	score
1	101	1	2025-01-10	88
2	102	2	2025-02-14	92
3	103	3	2025-01-30	95
4	104	4	2024-11-20	78
5	105	5	2025-01-15	85
- Results Tab:** Shows the results of the three queries. The first query returns 5 rows of employee data. The second query returns 3 rows of department data. The third query returns 5 rows of performance data.
- Message Tab:** Displays the message "No issues found".

--Top 3 score

select top 3

```

e.name,
d.departmentname,
p.score,
p.reviewdate
from performance p
join employee e on p.employeeid = e.employeeid
join department d on e.departmentid = d.departmentid
order by p.score desc;
```

```
    select top 3
        e.name,
        d.departmentname,
        p.score,
        p.reviewdate
    from performance p
    join employee e on p.employeeid = e.employeeid
    join department d on e.departmentid = d.departmentid
    order by p.score desc;

    -- Department Average Score
    select
        d.departmentname,
```

100 % No issues found

T-SQL Results Message

	name	departmentname	score	reviewdate
1	Ravi	Engineering	95	2025-01-30
2	Meena	HR	92	2025-02-14
3	Arjun	Engineering	88	2025-01-10

-- Department Average Score

```
select

    d.departmentname,

    avg(p.score) as avg_score

from performance p

join employee e on p.employeeid = e.employeeid

join department d on e.departmentid = d.departmentid

group by d.departmentname;
```

```
    p.reviewdate
    from performance p
    join employee e on p.employeeid = e.employeeid
    join department d on e.departmentid = d.departmentid
    order by p.score desc;

    -- Department Average Score
    select
        d.departmentname,
        avg(p.score) as avg_score
    from performance p
    join employee e on p.employeeid = e.employeeid
    join department d on e.departmentid = d.departmentid
    group by d.departmentname;
```

100 % No issues found

T-SQL Results Message

	departmentname	avg_score
1	Engineering	91
2	HR	88
3	Marketing	78

```
--Salary vs Score (Salary > avg and Score > 80)
```

```
select
```

```
    e.name,
```

```
    e.salary,
```

```
    p.score
```

```
from performance p
```

```
join employee e on p.employeeid = e.employeeid
```

```
where e.salary > (select avg(salary) from employee)
```

```
and p.score > 80;
```

The screenshot shows a SQL Server Management Studio (SSMS) window with the following details:

- Query Editor:** The main pane displays the T-SQL query for selecting employees based on salary and performance score.
- Status Bar:** Shows "100 %", "No issues found", and tabs for "T-SQL", "Results", and "Message".
- Results Pane:** Shows a table with two rows of data.

	name	salary	score
1	Ravi	85000	95
2	Kumar	90000	85

```
--New Joiners in 2024
```

```
select * from employee
```

```
where hiredate >= '2024-01-01' and hiredate <= '2024-12-31';
```

```
||| select * from employee  
||| where hiredate >= '2024-01-01' and hiredate <= '2024-12-31';
```

100 % No issues found

T-SQL Results Message

	employeeid	name	departmentid	salary	hiredate
1	2	Meena	1	60000	2024-07-01
2	3	Ravi	2	85000	2024-03-10
3	5	Kumar	1	90000	2024-08-20