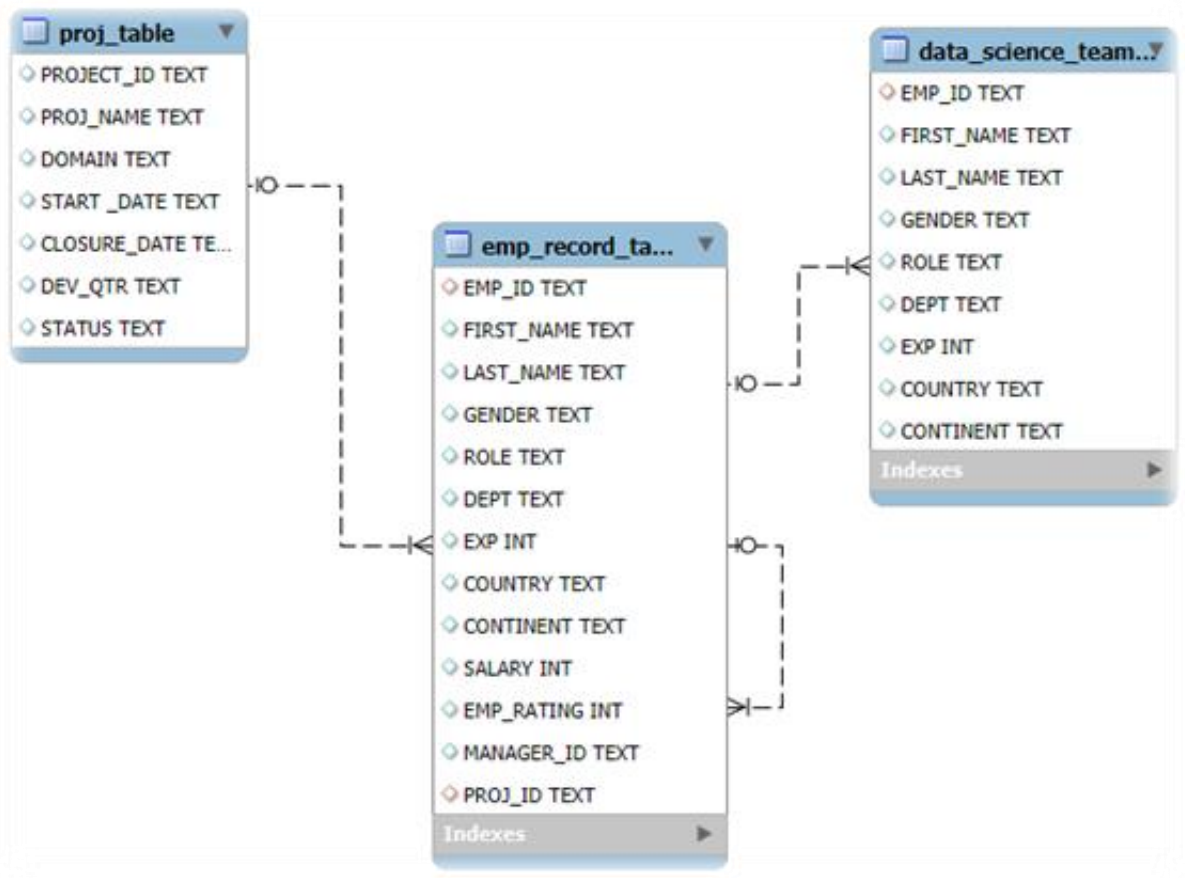
Project Query and Attachment

Q1. Create a database named employee, then import data_science_team.csv proj_table.csv and emp_record_table.csv into the employee database from the given resources.

Query:

```
CREATE database employee;
```

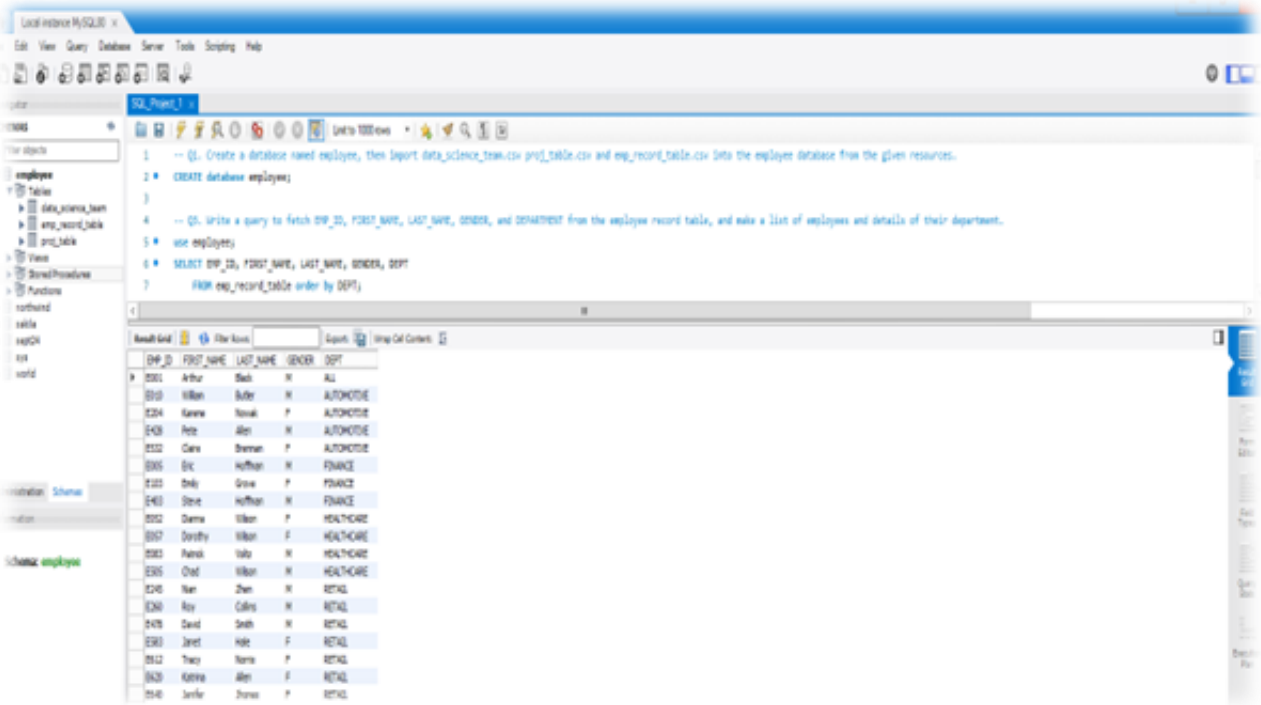
Q2. Create an ER diagram for the given **employee** database.



Q3. Write a query to fetch EMP_ID, FIRST_NAME, LAST_NAME, GENDER, and DEPARTMENT from the employee record table, and make a list of employees and details of their department.

Query:

```
SELECT EMP_ID, FIRST_NAME, LAST_NAME, GENDER, DEPT
    FROM emp_record_table order by DEPT;
```

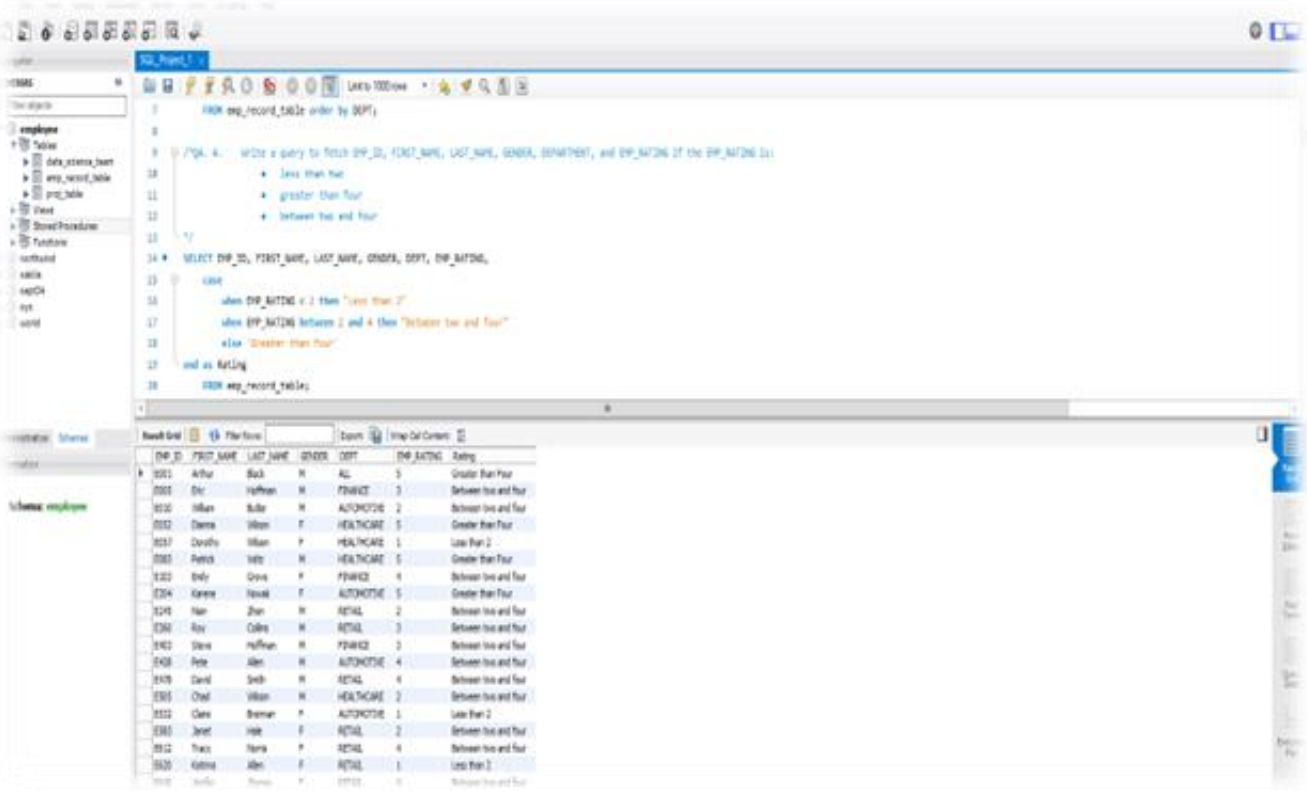


Q4. 4. Write a query to fetch EMP_ID, FIRST_NAME, LAST_NAME, GENDER, DEPARTMENT, and EMP_RATING if the EMP_RATING is:

- less than two
- greater than four
- between two and four

Query:

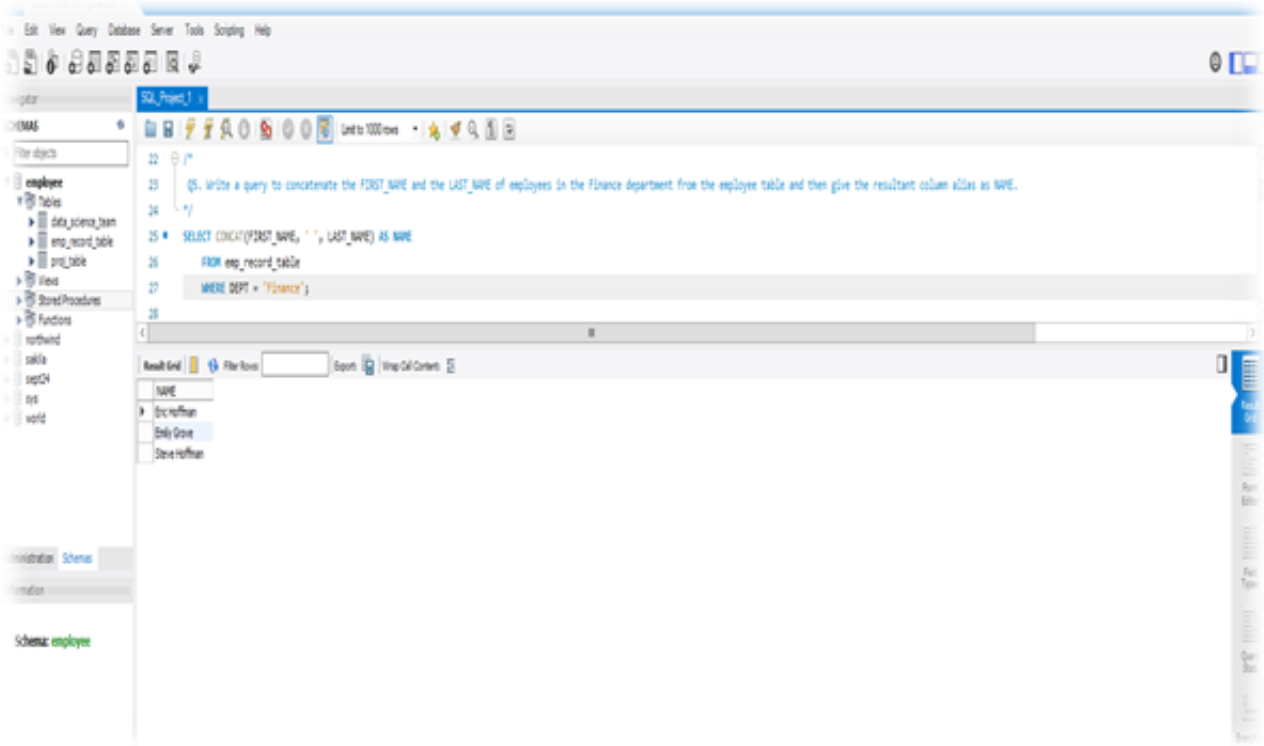
```
SELECT EMP_ID, FIRST_NAME, LAST_NAME, GENDER, DEPT,
EMP_RATING,
case
    when EMP_RATING < 2 then "Less than 2"
    when EMP_RATING between 2 and 4 then "Between two and four"
    else 'Greater than Four'
    end as Rating
FROM emp_record_table;
```



Q5. Write a query to concatenate the FIRST_NAME and the LAST_NAME of employees in the Finance department from the employee table and then give the resultant column alias as NAME.

Query:

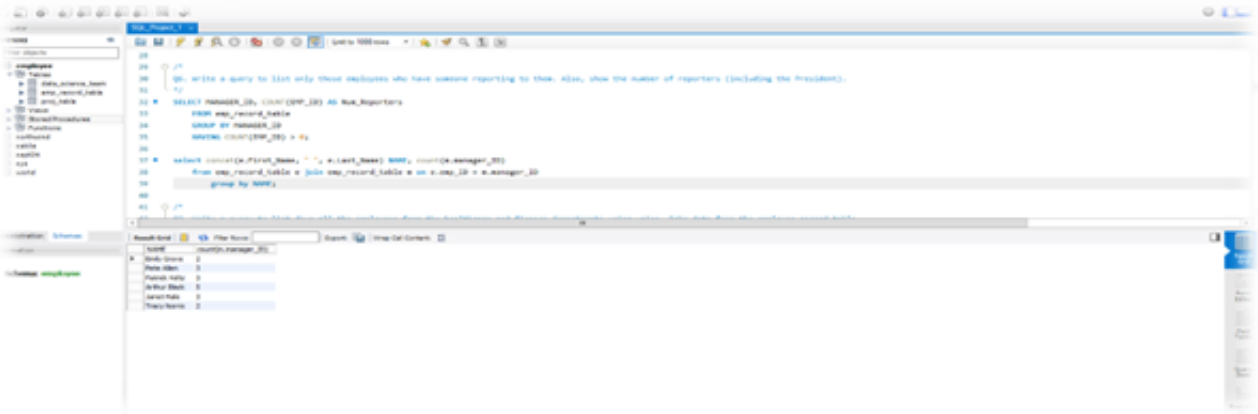
```
SELECT CONCAT(FIRST_NAME, ' ', LAST_NAME) AS NAME
FROM emp_record_table
WHERE DEPT = 'Finance';
```



Q6. Write a query to list only those employees who have someone reporting to them. Also, show the number of reporters (including the President)

Query:

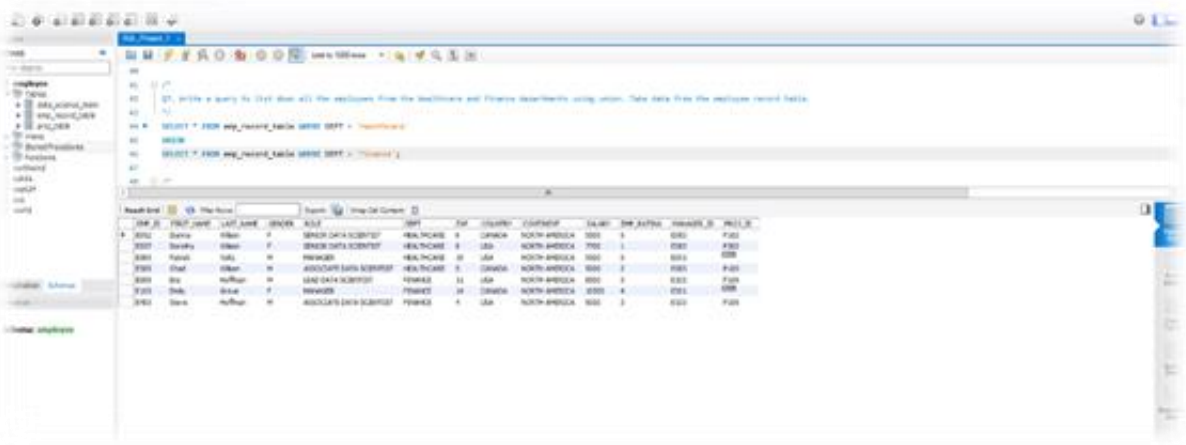
```
select concat(e.First_Name, " ", e.Last_Name) NAME, count(m.manager_ID)
    from emp_record_table e join emp_record_table m on e.emp_ID =
    m.manager_ID
    group by NAME;
```



Q7. Write a query to list down all the employees from the healthcare and finance departments using union. Take data from the employee record table.

Query:

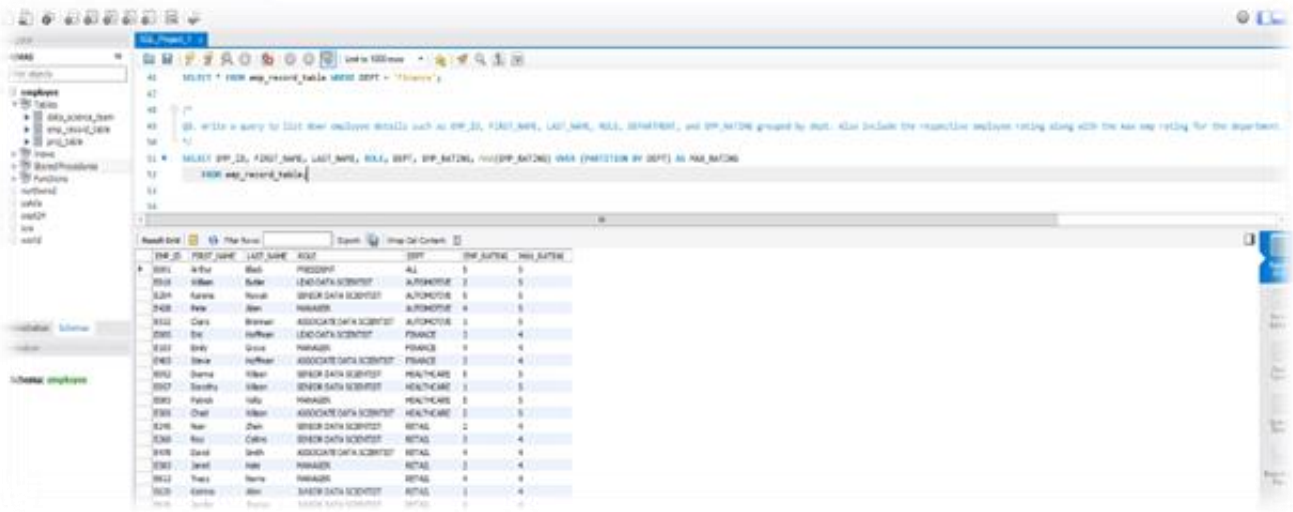
```
SELECT * FROM emp_record_table WHERE DEPT = 'Healthcare'
UNION
SELECT * FROM emp_record_table WHERE DEPT = 'Finance';
```



Q8. Write a query to list down employee details such as EMP_ID, FIRST_NAME, LAST_NAME, ROLE, DEPARTMENT, and EMP_RATING grouped by dept. Also include the respective employee rating along with the max emp rating for the department.

Query:

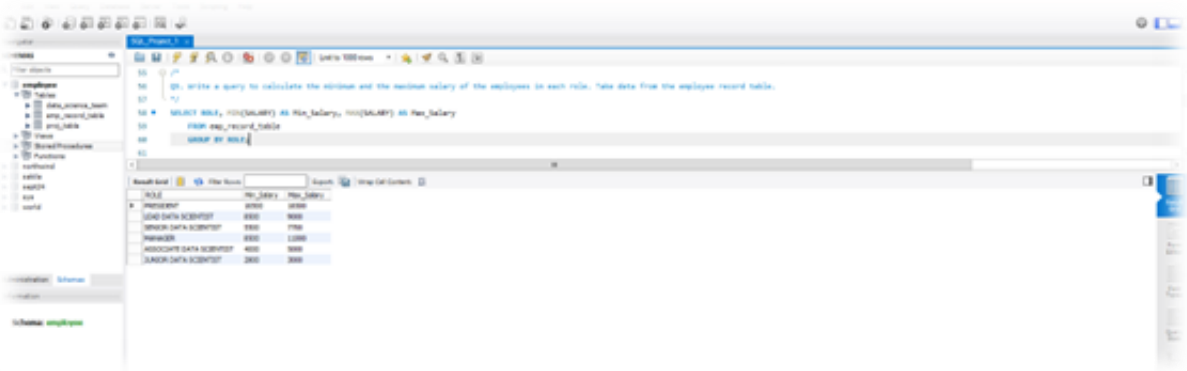
```
SELECT EMP_ID, FIRST_NAME, LAST_NAME, ROLE, DEPT,
EMP_RATING, MAX(EMP_RATING) OVER (PARTITION BY DEPT) AS
MAX_RATING

FROM emp_record_table;
```



Q9. Write a query to calculate the minimum and the maximum salary of the employees in each role. Take data from the employee record table.

Query:

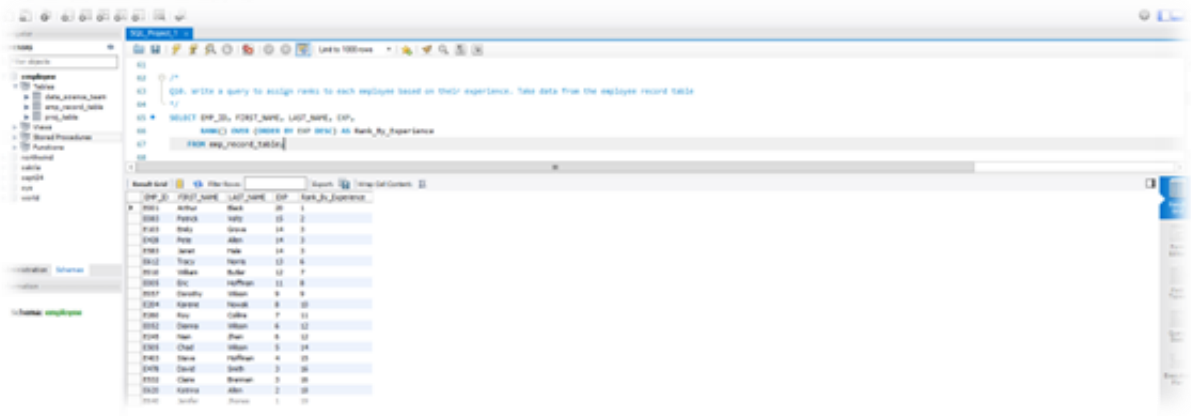
```
SELECT ROLE, MIN(SALARY) AS Min_Salary, MAX(SALARY) AS
Max_Salary FROM emp_record_table  GROUP BY ROLE;
```



Q10. Write a query to assign ranks to each employee based on their experience. Take data from the employee record table

Query:

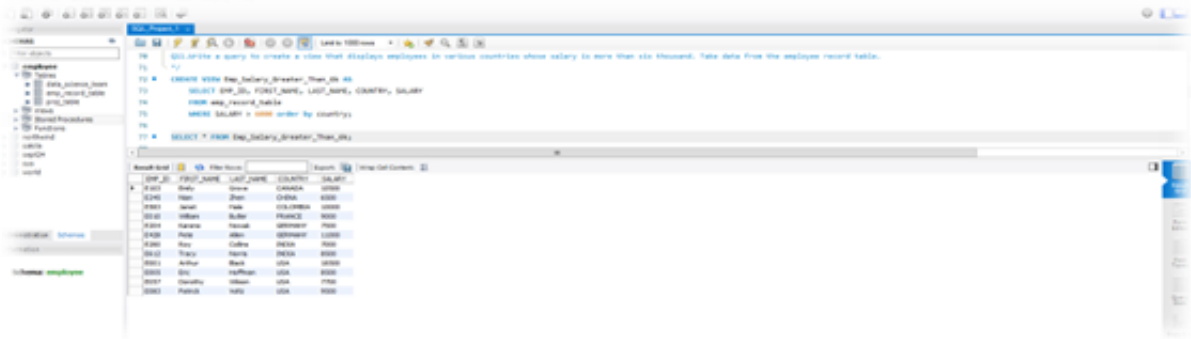
```
SELECT EMP_ID, FIRST_NAME, LAST_NAME, EXP,
RANK() OVER (ORDER BY EXP DESC) AS Rank_By_Experience
FROM emp_record_table;
```



Q11.Write a query to create a view that displays employees in various countries whose salary is more than six thousand. Take data from the employee record table.

Query:

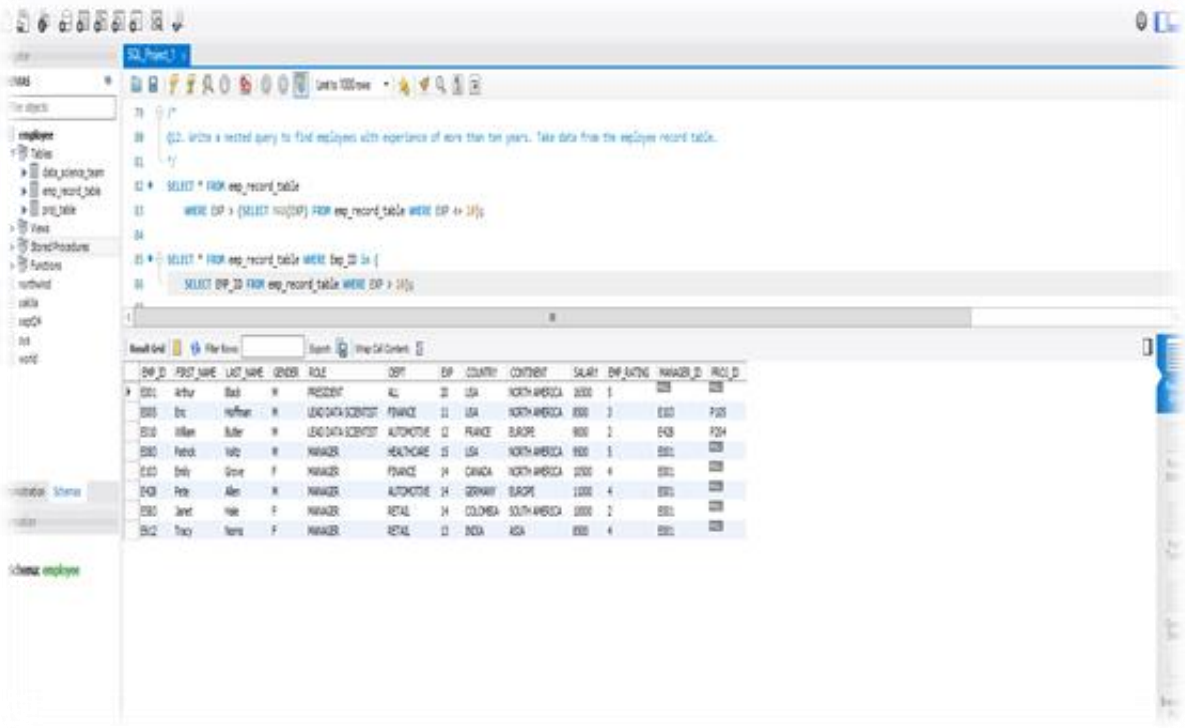
```
CREATE VIEW Emp_Salary_Greater_Than_6k AS
SELECT EMP_ID, FIRST_NAME, LAST_NAME, COUNTRY, SALARY
FROM emp_record_table
WHERE SALARY > 6000 order by country;
SELECT * FROM Emp_Salary_Greater_Than_6k;
```



Q12. Write a nested query to find employees with experience of more than ten years. Take data from the employee record table.

Query:

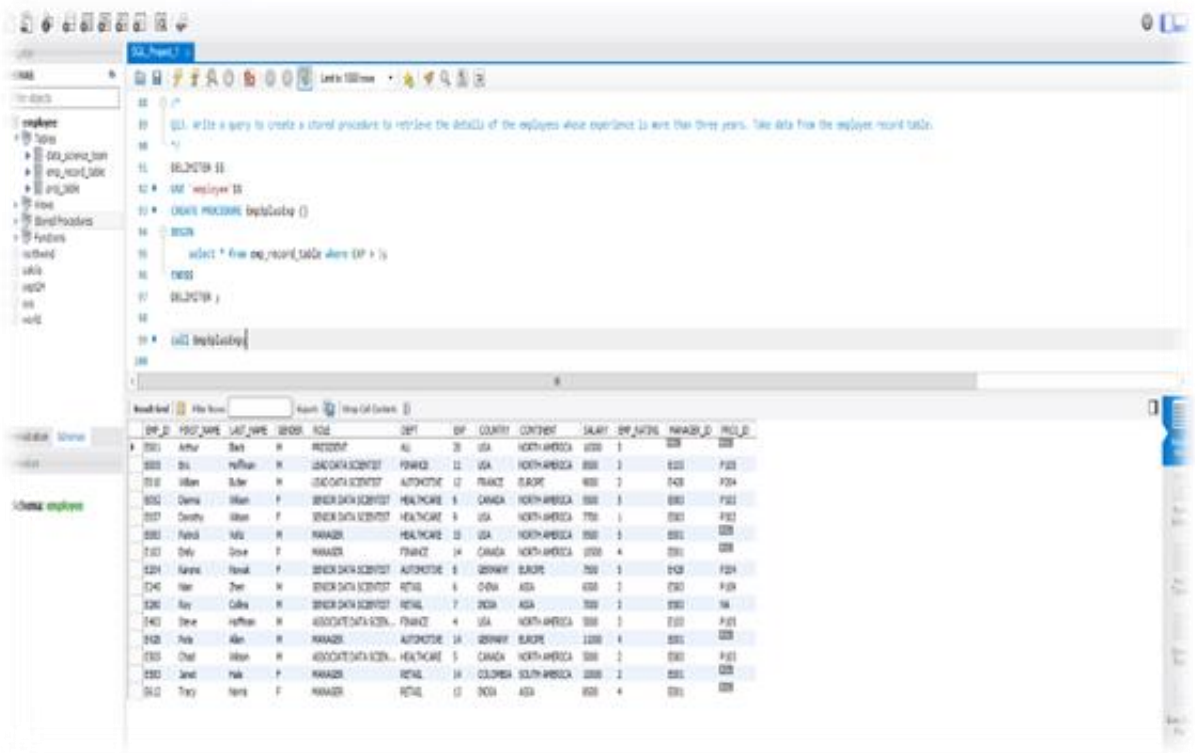
```
SELECT * FROM emp_record_table WHERE Emp_ID in (
        SELECT EMP_ID FROM emp_record_table WHERE EXP > 10);
```



Q13. Write a query to create a stored procedure to retrieve the details of the employees whose experience is more than three years. Take data from the employee record table.

Query:

```
DELIMITER $$

USE `employee`$$

CREATE PROCEDURE Emp3plusExp ()

BEGIN

        select * from emp_record_table where EXP > 3;

END$$

DELIMITER ;


call Emp3plusExp;
```



Q14. Write a query using stored functions in the project table to check whether the job profile assigned to each employee in the data science team matches the organization's set standard.

The standard being:

For an employee with experience less than or equal to 2 years assign 'JUNIOR DATA SCIENTIST',

For an employee with the experience of 2 to 5 years assign 'ASSOCIATE DATA SCIENTIST',

For an employee with the experience of 5 to 10 years assign 'SENIOR DATA SCIENTIST',

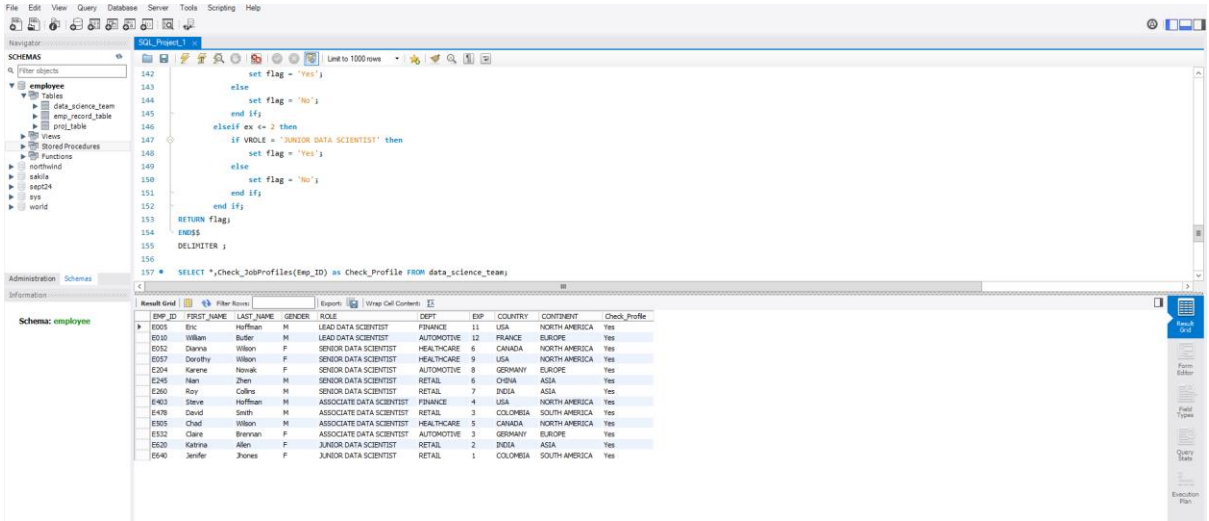
For an employee with the experience of 10 to 12 years assign 'LEAD DATA SCIENTIST',

For an employee with the experience of 12 to 16 years assign 'MANAGER'.

Query:

```
DELIMITER $$
USE `employee`$$
CREATE FUNCTION Check_JobProfiles (eid   varchar(5))
        RETURNS varchar(100)
   DETERMINISTIC
BEGIN
        declare ex int;
   declare r varchar(80);
   declare vrole varchar(100);
   declare flag varchar(10);
   select exp, ROLE into ex, VROLE from data_science_team  where emp_ID = eid;
                if ex > 12 and ex < 16 then
                        if VROLE = 'Manager' then
                                set flag = 'Yes';
                        else
                                set flag = 'No';
                        end if;
                        # set r = 'Manager';
```

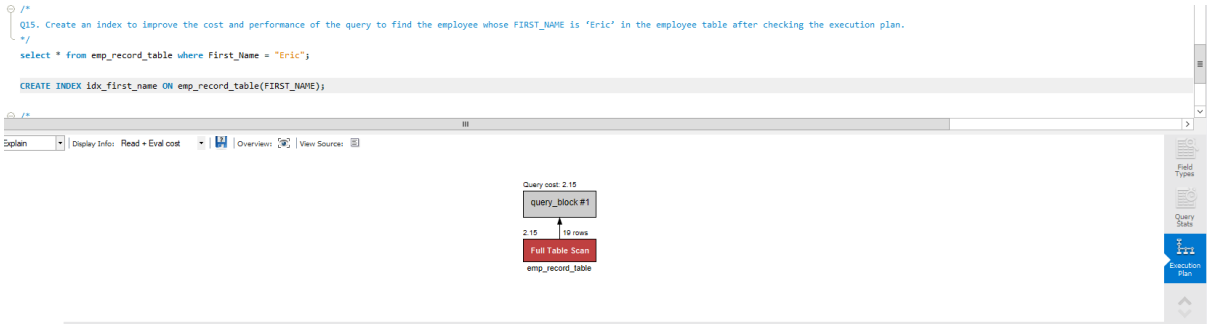
```
        elseif ex > 10 and ex <= 12 then
            if VROLE = 'LEAD DATA SCIENTIST' then
                set flag = 'Yes';
            else
                set flag = 'No';
            end if;
        elseif ex > 5 and ex <=10 then
            if VROLE = 'SENIOR DATA SCIENTIST' then
                set flag = 'Yes';
            else
                set flag = 'No';
            end if;
        elseif ex > 2 and ex <=5 then
            if VROLE = 'ASSOCIATE DATA SCIENTIST' then
                set flag = 'Yes';
            else
                set flag = 'No';
            end if;
        elseif ex <= 2 then
            if VROLE = 'JUNIOR DATA SCIENTIST' then
                set flag = 'Yes';
            else
                set flag = 'No';
            end if;
        end if;
RETURN flag;
END$$
DELIMITER ;


SELECT *,Check_JobProfiles(Emp_ID) as Check_Profile FROM
data_science_team;
```

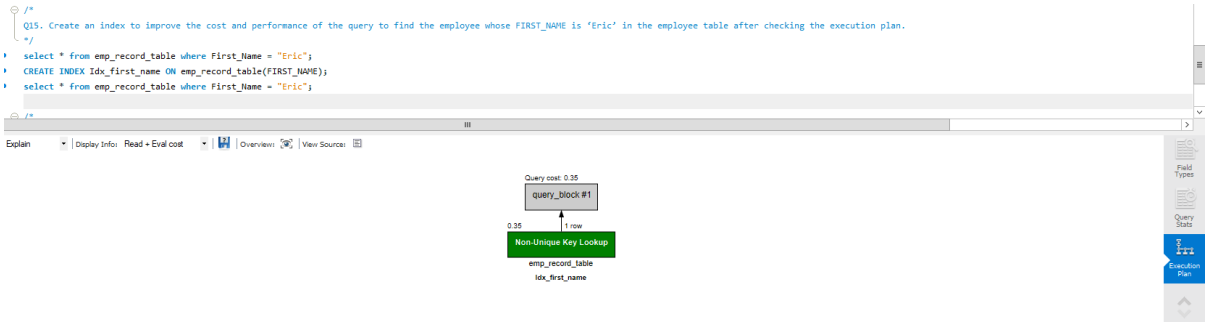


Q15. Create an index to improve the cost and performance of the query to find the employee whose FIRST_NAME is ‘Eric’ in the employee table after checking the execution plan.

Query:

```
select * from emp_record_table where First_Name = "Eric";

CREATE INDEX Idx_first_name ON emp_record_table(FIRST_NAME);

select * from emp_record_table where First_Name = "Eric";
```

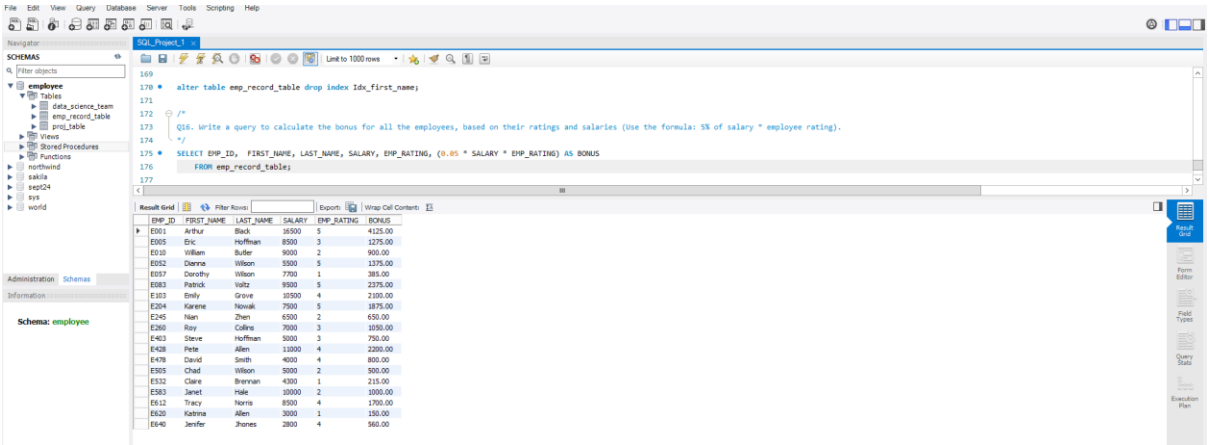




Q16. Write a query to calculate the bonus for all the employees, based on their ratings and salaries (Use the formula: 5% of salary * employee rating).

Query:

```
SELECT EMP_ID,  FIRST_NAME, LAST_NAME, SALARY, EMP_RATING, (0.05 *
SALARY * EMP_RATING) AS BONUS

        FROM emp_record_table;
```



Q17. Write a query to calculate the average salary distribution based on the continent and country. Take data from the employee record table.

Query:

```
SELECT CONTINENT, COUNTRY, AVG(SALARY) AS Avg_Salary

        FROM emp_record_table

        GROUP BY CONTINENT, COUNTRY with rollup

   order by CONTINENT, COUNTRY;
```